

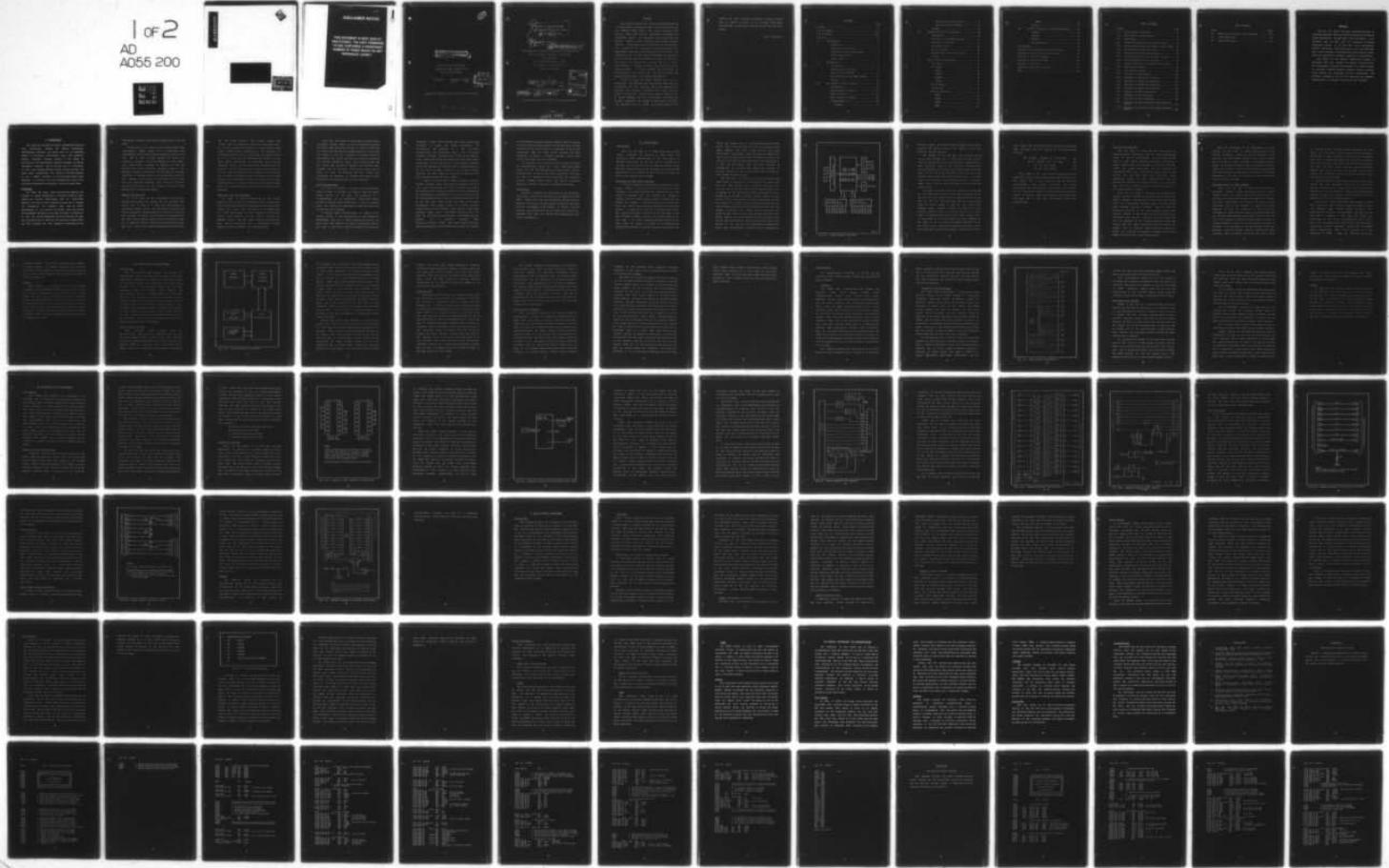
AD-A055 200 AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OHIO SCH--ETC F/G 14/2
DESIGN OF A MICROPROCESSOR BASED SYSTEM FOR TESTING OF THE NCR --ETC(U)
MAR 78 J W ROBERTSON

UNCLASSIFIED

AFIT/GE/EE/78-7

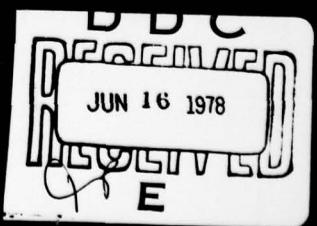
NL

1 of 2
AD
A055 200



ED

ADA 055200



DISCLAIMER NOTICE

**THIS DOCUMENT IS BEST QUALITY
PRACTICABLE. THE COPY FURNISHED
TO DDC CONTAINED A SIGNIFICANT
NUMBER OF PAGES WHICH DO NOT
REPRODUCE LEGIBLY.**

1

THIS DOCUMENT IS BEST QUALITY PRACTICABLE.
THE COPY FURNISHED TO DDC CONTAINED A
SIGNIFICANT NUMBER OF PAGES WHICH DO NOT
REPRODUCE LEGIBLY.

DESIGN OF A MICROPROCESSOR
BASED SYSTEM FOR TESTING OF THE
NCR 2051 MNOS MEMORY

GE/EE/78-7

Joel W. Robertson
Captain USAF



Approved for public release; distribution unlimited

78 06 13 038

DESIGN OF A MICROPROCESSOR

BASED SYSTEM FOR TESTING OF THE

NCR 2951 MNOS MEMORY

DESIGN OF A MICROPROCESSOR

BASED SYSTEM FOR TESTING OF THE

NCR 2951 MNOS MEMORY

Masters THESIS

THESES

147 AFIT/GE/EE/78-7

Presented to the Faculty of the School of Engineering
of the Air Force Institute of Technology

Air University

in Partial Fulfillment of the

Requirements for the Degree of

Master of Science

by

Joel W. Robertson B.S.E.E.

CAPTAIN

USAF

Graduate Electrical Engineering

(12) 252p.

(17) 17 Mar. 1978

ACCESSION FOR	
NTIS	White Section <input checked="" type="checkbox"/>
DOC	Buff Section <input type="checkbox"/>
UNANNOUNCED	<input type="checkbox"/>
JUSTIFICATION	<input type="checkbox"/>
BY _____	
DISTRIBUTION/AVAILABILITY CODES	
BISL	AVAIL. END OR SPECIAL
A	23 E.S.

Approved for public release; distribution unlimited

1473

012 225

LB

Preface

This report summarizes the design and implementation of a microprocessor based system that will perform testing of the NCR-2051 MNOS memory. The report discusses both hardware design and software development (including the development process and associated design decisions) and provides complete source listings of all assembly language software. This test system is to be used for two purposes: (1) a model for an acceptance testing system and (2) a programmable laboratory test facility for the NCR-2051. The techniques used for this system development are applicable to any microprocessor based memory test system development. The report is written for a person who has a basic understanding of digital circuits and some knowledge of microprocessors. A person without any background in microprocessors can operate the test system using the operating manual that is provided in Appendix C.

I wish to thank my thesis sponsor, Dr. Fritz L. Schuermeyer (research physicist with the Air Force Avionics Laboratory), for his technical advice and cooperation in providing the necessary components with which the test system was realized. I also wish to thank Dr. Gary Lamont, my thesis advisor, for his professional assistance and guidance throughout the project. I would like to extend a very important note of thanks to my wife (Pam) and son

(Scott) for their patience and support. Finally, I would like to express my thanks to my personal M6800 based microcomputer system which allowed me to edit and type this report.

Joel W. Robertson

Contents

	Page
Preface	ii
List of Figures	vii
List of Tables	viii
Abstract	ix
I. Introduction	1
Background	1
Statement of Objective	2
Constraints and Test Philosophy	3
System Configuration	4
Organization	6
II. MNOS Memories	7
Introduction	7
Description of operation	7
Application of NCR 2051	12
Test requirements for MNOS memories	13
Summary	15
III. Microcomputer system overview	16
Introduction	16
Microprocessor selection	16
Video terminal	19
Audio Cassette Interface	20
Microprocessor	23
Hardware	23

Memory Map of Microprocessor	24
Operating System (MIKbug)	26
Summary	28
IV. Development of System Hardware	29
Introduction	29
Parallel I/O Port Configuration	29
Interface to NCR 2051	31
Line Printer Port	40
Serial I/O Port	42
Microprocessor Board Modification	42
Summary	44
V. System Software Development	47
Introduction	47
Algorithms	48
MARCH	48
MASEST	49
WAKPAT	50
GALPAT	51
System Monitor	53
Test Monitor	56
General Subroutines	61
WINIT	61
WRITE	61
RINIT	62
READ	62

INDEX	63
Summary	63
VI. Results, Conclusions, and Recommendations . .	64
Results	64
Hardware	65
Software	66
Recommendations	66
Bibliography	68
Appendix A: System Monitor Software Listing	69
Appendix B: Test System Software Listing	79
Appendix C: Operator's Manual	101
Appendix D: Pulse Width Table	117
Appendix E: Flowcharts	119
Appendix F: NCR 2051 Data Sheets	131
VITA	138

List of Figures

Figure	Page
2-1 Block Diagram of NCR-2051	9
3-1 Block Diagram of Test System	17
3-2 Memory Map	25
4-1 Headers for MP-L Parallel Interface Cards	32
4-2 Schematic Diagram for MUT Negative Power Supply	34
4-3 Block Diagram of MUT Interface	37
4-4 Schematic Diagram of MUT Interface	39
4-5 Schematic Diagram of Line Printer Interface	41
4-6 Wiring Diagram of Serial I/O Port	43
4-7 Schematic Diagram of CPU Board Modifications	45
5-1 Test Monitor Display	58
C-1 Test Monitor Display	113
E-1 Flowchart of Test Monitor	120
E-2 Flowchart of MARCH Test Algorithm	121
E-3 Flowchart of MASEST Test Algorithm	122
E-4 Flowchart of WAKPAT Test Algorithm	124
E-5 Flowchart of GALPAT Test Algorithm	125
E-6 Flowchart of BCKGND Subroutine	126
E-7 Flowchart of READ Subroutine	127
E-8 Flowchart of WRITE Subroutine	128
E-9 Flowchart of Operating System Input Character Routine	129
E-10 Flowchart of Operating System Output Character Routine	130

List of Tables

Table	Page
C1 System Initialization Switch Settings	114
C2 Pulse Width Table	115
D1 Pulse Width Table	118

Abstract

The U.S. Air Force has begun using MNOS memories in ultrahigh frequency radios to provide nonvolatile storage of preset frequencies. The NCR 2051 is one such memory. It is necessary that the USAF have the capability to perform acceptance testing of the NCR 2051 and to economically perform laboratory tests which may be very time consuming. This report develops a microprocessor-based computer system which will provide the necessary capabilities economically.

The design of the Motorola M6800-based system is presented with both hardware and software considerations. The development decisions are discussed and a user's manual is provided. Complete assembly language software listings, which realize the acceptance testing requirements, are included. Flowcharts for all test algorithms and schematic diagrams for all interface circuits are also provided.

I. INTRODUCTION

The need for storage of digital information arises as more electronics systems use digital processing. Semiconductor memories are being used in an increasing number of processing applications since they provide a compact low-power storage device. As the number of applications for semiconductor memories expands, the memory complexity increases (due to the fact that it is desirable to have more storage capacity, higher storage density, and lower power consumption). The problem of verifying whether or not a memory performs all its functions becomes more difficult to resolve. It is this problem that leads to the following investigation concerning a specific memory type.

Background

The USAF has begun using semiconductor memories for storage of preset frequencies in ultrahigh frequency (UHF) radios for airborne applications (Ref 10). Since these memories are used to store information which may be changed (as frequencies are changed) they must be easily programmable in the aircraft (in flight). They must retain the programmed information even when the power is turned off so that the preset frequencies will be available the next time aircraft power is restored. One particular memory which has been selected for this purpose is designated by its

manufacturer (National Cash Register Corporation) as the NCR 2051.

The NCR 2051 is a 32 word by 16 bit metal-nitride-oxide semiconductor (MNOS) memory. It is electrically word programmable and will retain stored information for up to 10 years (Ref 9) with no power applied (in or out of an electrical circuit). Due to the complexity of this memory, very sophisticated equipment is required to test the NCR 2051 for proper operation. Equipment (such as the Macrodata MD-104 memory tester (Ref 4)) which is currently available to perform testing of this integrated circuit is very expensive and requires a skilled operator with a high level of technical knowledge. This cost and technical expertise combination was the primary motivation which prompted this investigative effort.

Statement of Objective

It is the goal of the investigation to develop a microprocessor-based test system for the NCR 2051. This system will be used as a model for constructing an acceptance testing system. It will provide the Air Force with an inexpensive method (compared to the cost of existing test systems) of testing and evaluating the performance of the MNOS memories before accepting them and utilizing them in an operating avionics system. The resulting test system must also reduce the technical expertise requirements for

the test system operator. This provides another cost savings. The system which is produced by this effort will, in addition, provide the capability to economically perform tests, (which require dedication of the equipment for extended periods of time), on the memory. The software which is to be written for the system must be flexible so as to allow simple modification to accomodate more elaborate laboratory tests. A user's manual (See Appendix C) must be provided. The test monitor will be written (See Appendix B) so that very few references to the manual will be required and prompting information will be displayed on the system console. The system must be capable of performing each test automatically once a test is initiated (the operator simply types the instructions that select the test which is to be executed).

Constraints and Test Philosophy

In order to provide flexibility to the testing procedures, the testing of the NCR 2051 will be accomplished using an operator-controlled microcomputer system which will control all signals that are applied to the memory under test (MUT). The operator will have control over write/erase timing, test patterns, and test program selection by keyboard entries from the system console. Read timing is also operator controlled, but is accomplished through hardware (switching) instead of a keyboard entry.

Since the test system is to be used as a model for an acceptance testing system, it will be developed to perform only those tests which are necessary to assure the USAF that the device will perform its intended function in the system into which it will be installed. No attempt is made to characterize the device with the test system, as measuring all possible parameters would require too much time. The cost would then override the benefits to be derived. Error messages will be displayed on the system console (in the event an error is detected) and will include such information as necessary to determine the exact location within the test sequence where the error occurred.

System Configuration

In order to provide a testing system which may be easily changed to perform any desired test, the system should consist of at least the following: (1) a microprocessor, (2) an alphanumeric communication device (such as a teletype video terminal etc.), and (3) a storage media. Any additional input/output device may add to the capabilities of the system.

A Motorola M6800 microprocessor was chosen as the central element in the system because of its treatment of Input/Output (I/O) as a memory read/write operation, its small size (the M6800 is an eight bit processor), and its low cost. A test system could be designed starting at the

integrated circuit level; however, due to the limited time available to finish the system development, the microcomputer was purchased as a kit. The basic microprocessor was constructed using a kit produced by Southwest Technical Products Corp. (SWTPC) of San Antonio, Texas (model MP-A) since the design of the kit seems almost tailor-made for this application. A video terminal produced by the same company was available from a previous project in the Air Force Avionics Laboratory so an audio cassette interface kit from SWTPC was also selected (based on cost and compatibility) for the basic system.

A Hewlett Packard model 9820A calculator system was already available which had several peripheral devices. A test system with greater flexibility and reduced cost could be provided if the calculator system were to be connected to the microcomputer to allow the use of these peripherals with the microcomputer system. Since the calculator system is not in use full time, both systems can make use of these input/output (I/O) devices which are: a digital cassette storage system, an X-Y plotter, a line printer, a high-speed papertape reader, and a high-speed papertape punch. Two types of interfaces were developed to interconnect the two systems: (1) a parallel interface (with transistor-transistor logic (TTL) signal levels (Ref 11)) connected directly to the HP 9866A line printer (the printer

is disconnected from the calculator system) and (2) a serial interface (RS-232) connecting the test system into the 9820A controller which allows data to be transferred to any of the peripherals (Ref 2). This serial interface was also configured to accept a standard teletype, using a 20ma current loop, which was also available.

An interface was designed and constructed to connect the microprocessor system to a test socket which would accept the NCR 2051 MNOS memory and which would allow the microprocessor to control the signals applied to the memory in order to implement the desired testing. This interface utilizes standard SN7400 series TTL devices to perform buffering, latching, inverting, and delaying functions.

Organization

Chapter II. provides the description of the operations, applications, and testing procedures of the MNOS memory (NCR 2051). Chapter III. describes the overall testing system as designed, developed, and implemented. Chapter IV. shows the hardware development while Chapter V. follows with the software implementation of the test algorithms. Chapter VI. concludes with some test results and recommendations for further development.

II. MNOS MEMORIES

Introduction

Since the NCR 2051 is an MNOS memory device, this chapter describes the operation of MNOS memories to facilitate reader understanding of the test system. A detailed description of the NCR 2051 is included and is followed by a discussion of an Air Force application of the NCR 2051. Several factors to be considered when testing MNOS memories are also discussed, in order to help the reader see why the test routines were selected.

Description of MNOS Memory Operation

A metal-nitride-oxide semiconductor (MNOS) memory uses charge storage at the oxide-nitride interface at the gate insulator of the MNOS transistor for retention of data. The charge is tunneled into this interface by applying a negative writing voltage and the charge is trapped when the voltage is removed. A positive voltage may be applied to drain the charge thus erasing the data that was previously written. A memory cell is made up of two MNOS transistors. The presence of charge at the gate of one transistor of the pair determines whether a one or a zero voltage level (one = +5v, zero = 0v) will be detected when the memory cell is read. Since the charge is trapped at the oxide-nitride interface and requires a positive voltage applied before the

charge can escape, this is a nonvolatile type of storage device which may retain data in excess of 10 years with no power applied. The MNOS device differs from other electrically programmable read only memories (EPROM's) in that it can be electrically erased (as opposed to erasure by exposure to ultraviolet light radiation) and reprogrammed on a word by word basis. This erase/write function is accomplished by applying the appropriate positive or negative voltage to the desired memory cells to tunnel or drain charge from those cells.

The NCR 2051 is a 512 bit electrically alterable ROM (MNOS) which is organized as a 32 word by 16 bit memory (See Fig 2-1). MNOS devices have a limited read/write cycle lifetime and the NCR 2051 may be erased and rewritten up to a maximum of 10^6 times (Ref 9). This limitation is due to degradation of the MNOS transistor. It may, however, be read up to 2×10^{11} times before a refresh is necessary. Reading the memory cell allows the rate of charge leakage from the oxide-nitride interface to increase during the time the read voltage is present. This charge must be replaced before enough has been lost to cause incorrect data retrieval. This memory requires two power supplies for its operation (-29v and +5v when interfaced to transistor-transistor logic (TTL) devices or -24v and +10v when interfaced to complementary metal oxide semiconductors (CMOS)). With the exception of

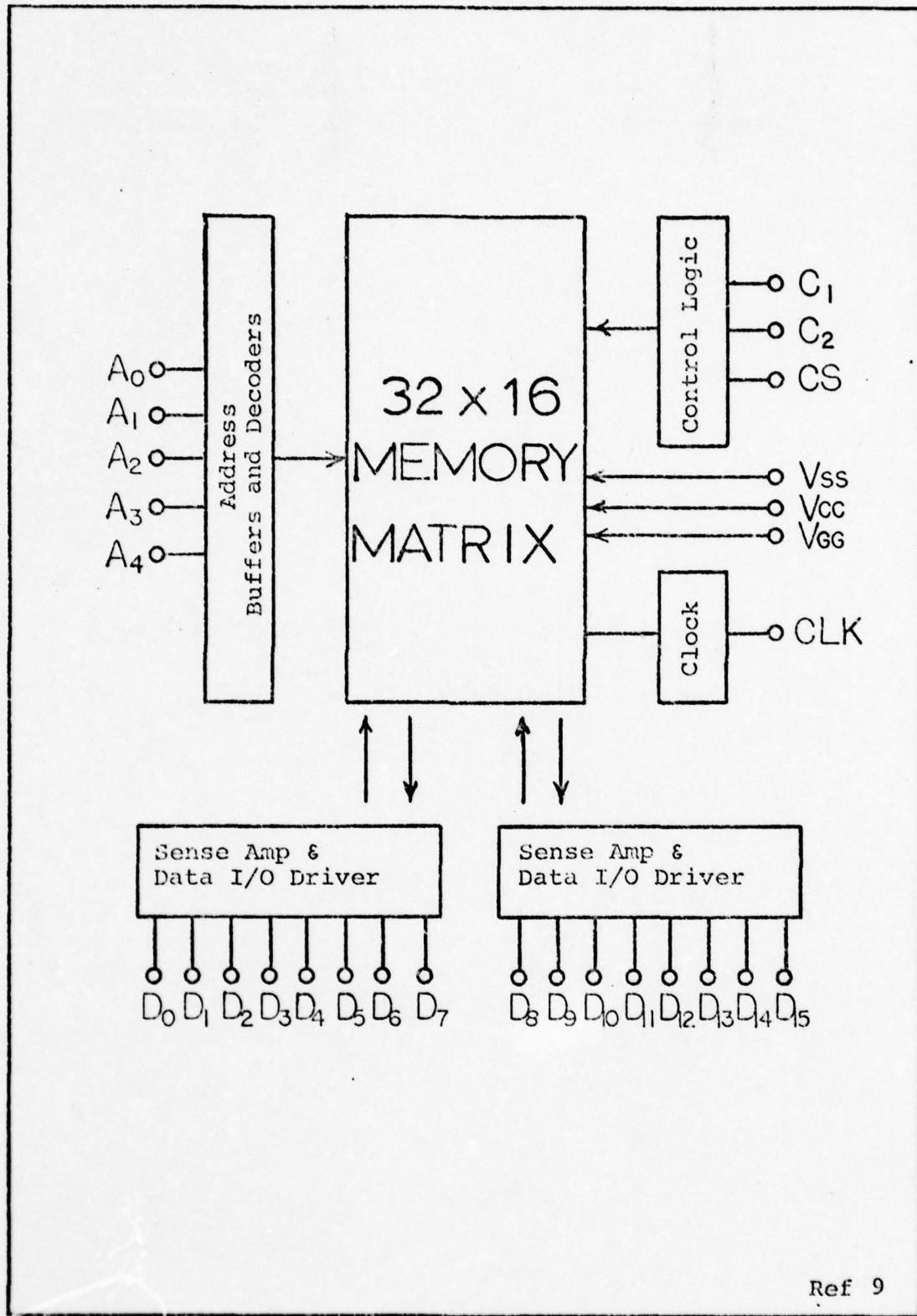


Fig. 2-1. Block Diagram of NCR-2051

Ref 9

the power supply, all I/O and control signals operate within standard TTL voltage levels (also capable of operating directly with CMOS (Ref 9)).

The outputs of the NCR 2051 have an access time of two μ s (See Appendix F). A clock pulse of 2-20 μ s pulse width is required for a read operation but is optional for a write operation. Data is not valid for a read operation until approximately 2 μ s after the falling edge of the clock pulse (Ref 9). The total word access time is the sum of the minimum clock pulse width and the access time - 4 μ s. This memory was obviously not intended for use as a read/write (RAM) memory since it requires 40-200 ms each to erase and rewrite data.

The mode in which the device is operating is controlled by the control logic that is in the same integrated circuit (See Fig 2-1). This control logic circuitry has three inputs (CS, C1, and C2). The chip select (CS) input essentially turns the entire device on and off (the device still consumes power, however), leaving the outputs in an open circuit condition when it is turned off (i.e. CS = logic zero). The remaining two inputs select either a read, write, or erase mode. If C1 is high (logic one) then regardless of the state of C2 a read mode is selected. However if C1 is low (logic zero) then C2 will determine the mode. If C2 = low (logic zero) then the write mode is selected and if C2 =

high (logic one) the erase mode is selected. The following logic equations show the function of the control logic circuits.

$$\overline{CS} \cdot (C1 + \overline{C1}) \cdot (C2 + \overline{C2}) = \overline{CS} = \text{Device off} \quad (1)$$

$$CS \cdot C1 \cdot (C2 + \overline{C2}) = CS \cdot C1 = \text{Read} \quad (2)$$

$$CS \cdot \overline{C1} \cdot C2 = \text{Erase} \quad (3)$$

$$CS \cdot \overline{C1} \cdot \overline{C2} = \text{Write} \quad (4)$$

Since there is no requirement for separation of the application of the clock signal from any other input signal, all inputs for a given operation may be initiated simultaneously. This greatly simplifies the design requirements for an interface to this memory. If separation of the signals were required, additional hardware would be needed to provide delays between each of the signals when the delay time is less than the resolution time of the processor I/O.

Application of NCR 2051

The Air Force is using the NCR 2051 memory in frequency-preset applications (Ref 10). Ultrahigh frequency radios in airborne installations are required by the Air Force to have twenty channels which may be selected by a single control knob. These channels are preset by the pilot and/or the ground maintenance crew to correspond to frequencies which are in use by the control tower, ground control, etc. and other often used frequencies. These frequencies change as the aircraft moves from one geographical location to another; therefore, it is important that the pilot be able to assign a new frequency to any or all channels while in flight.

Based on the author's experience in aircraft radio repair, previous methods of changing preset frequencies usually included some physical movement of several sliding parts of a mechanical drum-type memory. These mechanical parts actuated electrical switches when the drum was positioned according to the desired preset channel. This required opening a door on the front of the control panel (sometimes even complete removal of the radio control box from its mounting in the aircraft instrument panel) and leaning down for close observation of the setting of the sliders. This is virtually impossible for the pilot of a single seat aircraft to accomplish in flight since he must devote his attention to flying the aircraft.

With the development of the MNOS memory, it is now possible for the pilot to simply tune the radio to the desired frequency, using the manual frequency select controls, set the channel selector to the channel to which the frequency is being assigned, and momentarily push a button to program the memory. Other semiconductor memories (RAM's for example) could be used for this purpose except that each time the aircraft power is removed, all the preset frequencies would have to be re-entered. This would not be practical and is not used.

Test Requirements for MNOS Memories

Since it is the goal of this effort to develop an acceptance testing system (acceptance testing implies testing whether the device can be expected to work) it is necessary to evaluate only certain parameters of the memory. The basic requirements are to see if the device is fully operational and can be expected to operate in a system designed to allow all parameters to fall within the manufacturer's specifications. Based upon previous work with the NCR 2050 (functionally equivalent to the NCR 2051) (Ref 10), it is necessary to perform pattern sensitivity tests to determine whether a device is to be accepted or rejected. In addition, it is desirable to test the retention capability. Retention tests are, however, very time consuming and would introduce a delay in putting the devices into service. Based

on previous positive retention study results (Ref 10), this effort will not pursue the development of retention tests. Simulation of data retained at the end of a given storage time may be achieved by writing the data with too short a write time (Ref 10). This capability will be provided since the write and erase pulse widths are under full software control and may be selected by the system operator. The system will be flexible and will accomodate retention tests with simple software changes.

Pattern sensitivity tests perform repeated write, read, verify, and erase operations using one of several well defined sequences. This method of testing verifies full operation of all bits within the memory, and also tests all address decoding and sensing logic elements which are an integral part of the integrated circuit.

Due to the limited read/write cycle lifetime and the long write/erase times required, many test sequences which are effective when testing other read/write (RAM) semiconductor memories are impractical. They may take too much time to perform, or else they will degrade the performance of the device and effectively "wear it out". There are four widely used pattern sensitivity tests which collectively perform sufficient testing with negligible device degradation. These tests (referred to in the literature as MARCH - write and read/write forward and

backward, MASEST - alternating ones and zeroes, GALPAT - galloping pattern, and WAKPAT - walking pattern (Ref 10)) write data into the memory then read and verify that data in one of four well defined sequences and are described in detail in Chapter V.

Summary

MNOS memories store charge at the gate of an MNOS transistor. This nonvolatile storage may be retained for up to 10 years without power. The Air Force is using an MNOS memory to store preset frequency information in airborne radios. Pattern sensitivity tests are the most effective way of testing MNOS memories to verify proper operation because these tests verify the proper operation of all memory bit positions as well as all address decoding and logic sensing circuitry under worst case timing conditions (rapid access of adjacent cells etc.).

III. MICROCOMPUTER SYSTEM OVERVIEW

Introduction

The test system must perform each desired test automatically once a test has been selected. It must give information concerning the failure or successful operation of the device under test. Therefore, the test system consists of a microprocessor system, its associated equipment, and an interface which connects the microprocessor system to the MNOS memory to be tested (See Fig 3-1). The interface is discussed in chapter IV. This chapter describes the selection of the microprocessor which is to be used in the test system. A system overview is then presented which describes several kits which were purchased to form the basic microprocessor system. A video terminal, an audio cassette interface, and the microprocessor itself are described. Both hardware and software descriptions of the basic microprocessor are included.

Microprocessor Selection

After examining several available eight bit microprocessors (such as the Intel 8080, the Zilog Z80, and the Motorola M6800), their instruction sets, their treatment of I/O, the software development systems available etc., the Motorola M6800 was selected. Due to the higher cost of larger (more than eight bits) microprocessors, only eight

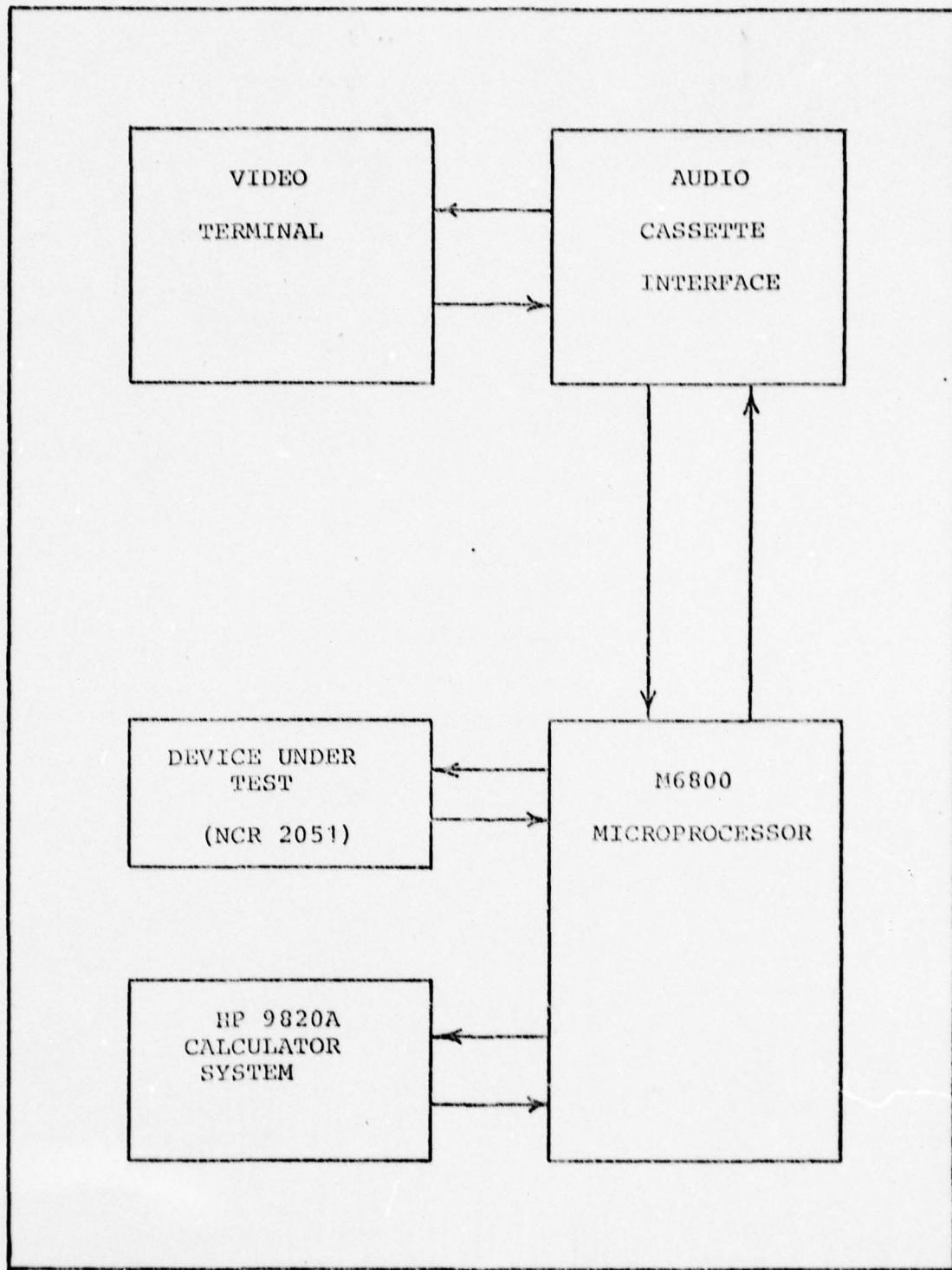


Fig. 3-1. Block Diagram of Test System

bit machines were considered. Any of the microprocessors considered could have been used for the test system as there is very little difference in their capabilities (except speed). One reason for the selection of the M6800 is that I/O is treated as a load or store operation to a memory address. Since the primary function of the processor will utilize I/O operations very frequently, this is an important feature. Also, a cross-assembler and simulator for the M6800, which operates on the CDC 6600 computer system, is available at the Air Force Institute of Technology (AFIT), so there is ample software development capability. Moreover, there are several persons in the Air Force Avionics Laboratory who have considerable experience using the M6800 system. Thus a local source of knowledge/experience is available in the event difficulties are encountered with the system development.

Rather than design and fabricate a microprocessor system from scratch, it seems very practical to purchase a kit since there are several inexpensive microprocessor kits available (there were no preassembled boards available for the M6800). There were two kits available (at the time the microprocessor was selected) which use the M6800. One kit, distributed by Altair Computer Centers is very well constructed but is rather small (physically). Since the Altair kit is very compact, it is not well suited for adding

several I/O ports. The system produced by Southwest Technical Products Corporation (SWTPC) is ideally suited for this application (it has space for several I/O ports provided which can be used to connect to the memory under test). Since the SWTPC system was selected, detailed descriptions of the SWTPC kits follow (see Chapter I. for kit selection information):

Video Terminal

Rather than use a teletype or switch panel on the front of a processor, a Video terminal was selected to be used by the operator to communicate with the system (to give commands and read data). A wide variety of video terminals is available with a multitude of options. The SWTPC CT-1024 video terminal was selected because it is available at no cost (the Avionics Lab had previously purchased it) and is compatible with the other kits purchased for the system. Any type of terminal using ASCII encoding, even a teletype, is acceptable. However, the SWTPC terminal provides decoding of user selectable control characters that may be used to control the audio cassette interface (described later in this chapter) which in turn will control the motor of the tape recorder. This will give the system an automatic start/stop feature for cassette tape operations. Although this is not a mandatory feature for system operation, it does make system operation simpler.

The CT-1024 terminal was designed for a 16 line by 32 character/line page with two pages of memory. In order to make the display of data more meaningful, several modifications were incorporated into the terminal (these modifications are not necessary if the Ct-64 terminal is purchased). Since 32 characters per line will not display a 40 character record block (MIKbug punch format), the screen format has been changed to a single page with 16 lines of 64 characters each with automatic scrolling (each line is printed at the bottom of the page and a line feed causes the entire page to be shifted up one line - similar to the operation of a typewriter).

Audio Cassette Interface

There are several methods used for saving programs for microcomputers (such as the floppy disc, digital cassettes, paper tape, and audio cassettes). Since one of the major guidelines of this effort is that the system is to be low cost, and there is no requirement for large amounts of data storage/retrieval, the audio cassette storage system was selected because it is the cheapest method considered and it will work for this application. The audio cassette interface provides a very inexpensive method of saving programs for reloading into the system after power is applied the next time. It is connected between the processor and the video terminal in a series fashion. Command signals from the

terminal to the interface allow automatic start/stop operation of the tape recorder through the motor control jack on the tape recorder.

The SWTPC AC-30 cassette interface was selected because of its low price (\$69.95) and ease of integration into the overall system design (it connects in series between the video terminal and the processor and no new software is required for its use). It is also the only kit currently available which will connect directly to an M6800 system. Other cassette interface kits such as the Tarbell Cassette Interface (Tarbell Electronics of Carson, California) plug into the S-100 buss structure of the 8080 processor and are therefore not useable with the M6800. This interface accepts data in RS-232 format from either the terminal (local mode) or the microprocessor (remote mode), depending upon the setting of the local/remote switch on the cassette interface front panel. This data is then converted into audio signals of 1200 or 2400 Hz for ones and zeros and is output to the microphone input of a single track audio cassette tape recorder. Recovery of information takes place in exactly the reverse order with the output coming from the headphone jack of the recorder. The data rate is 300 baud. The 16X clock signals (4800 Hz) for the terminal and the processor I/O port are routed through the interface. This allows the interface to use a self-clocking technique to generate the

clock signals while reading a cassette tape, thus enabling correct data recovery for a wide range of tape speed variation (This is a very important feature when utilizing inexpensive audio recorders since they have very poor motor speed controls).

Microprocessor

The microprocessor description is divided into the following three areas: hardware, memory map, and operating system (MIKbug).

Hardware

The SWTPC MP-A microprocessor kit includes the following items: power supply, chassis, system interconnecting circuit board (called mother board), processor circuit board (which contains the microprocessor and associated circuitry such as the clock, monitor ROM, etc.) , one four kbyte RAM memory board, and the control interface board (which connects to a serial device through an RS-232 or 20 ma current loop). Additional kits purchased include three four kbyte RAM memory boards, three parallel interface cards, one serial interface card, and one EPROM card (The model RB-68-8 EPROM card is manufactured by Shifting Sands Microcomputer Products Corp. of Fairborn, Ohio and is plug-in compatible with the SWTPC system). The EPROM board allows all developed programs to be placed in electrically programmable read only memories (TMS 2708's) so that the program will not have to be re-entered each time it is to be used.

The complete microprocessor (with space for six memory boards and eight interface cards) is housed in an attractive

black anodized aluminum case which measures 15 1/2 in long by 15 1/2 in wide by 7 in high. The only controls provided are two push-button switches for power and reset (Ref 6). All other functions are controlled through communications with the CRT terminal.

Memory Map of Microprocessor

The SWTPC M6800 microcomputer is designed using an operating system called MIKbug (MIKbug is a registered trademark) which was written by Motorola. Since this operating system must be used to run the test system, it is described in detail later in this chapter. The MIKbug monitor occupies 512 bytes of memory, but due to only partial address decoding, (to save hardware costs) it responds to the upper eight kbyte block of memory (\$E000 to \$FFFF) (the "\$" prefix means that the number to which it is attached is a hexadecimal number) repeating itself every 512 bytes (See Fig 3-2). This monitor uses 128 bytes of scratch pad RAM located at \$A000 to \$A07F. It uses four addresses (\$8004-\$8007) for the I/O to the control terminal.

The microprocessor kit is provided with connections and decoding for seven additional I/O ports so that I/O occupies memory addresses \$8000-\$801F. Again, due to partial address decoding, the I/O ports respond to multiple addresses -- alternate 32 address blocks from \$8000 to \$9FFF (i.e.. repeats \$8000-\$801F, \$8040-\$805F, \$8080-\$809F, etc). The

Fig. 3-2. Memory Map of Microprocessor

scratch pad RAM rolls over throughout \$A000 to \$BFFF. The memory map of the system is shown in Fig 3-2.

This configuration allows the user to install a continuous run of memory from \$0000 to \$7FFF (32 kbytes). In addition, eight kbytes are available to the user from \$C000 to \$DFFF. Additional memory areas may be used but modifications to the existing circuit boards would be necessary to more fully decode the addresses of the system.

Operating System (MIKbug)

MIKbug is the name of a monitor program written by Motorola for the M6800 microprocessor. It is designed to use a serial communication loop with a terminal such as a CRT or a teletype. The data transfer actually occurs through a parallel interface adapter (PIA) (Ref 8) but the software converts the data to a serial mode using only one input and one output line of the PIA. MIKbug (Ref 1) allows the user to perform one of five functions with a single character input command from the keyboard. These commands and their functions are as follows:

(1) R Examine the contents of the stack. This actually examines the contents of seven memory locations (\$A043 to \$A049) which will be loaded into the condition code register (CC), the A accumulator (ACCA), the B accumulator (ACCB), the index register (X), and the program counter (PC) respectively when the go to user program command is given.

(2) G Go to user's program. This command causes a return from interrupt instruction to be executed. This loads the registers from the stack and begins execution at the address which was loaded into the program counter (from \$A048 and \$A049).

(3) M Memory examine or change function. This allows the user to examine any memory location and change its contents (if that location is RAM). This routine returns a question mark if the contents of the desired memory location did not change to the desired data.

(4) P Print/Punch contents of memory function. This outputs data between the beginning address (stored in \$A002/3) and the ending address (stored in \$A004/5) with the appropriate checksums and addressing information for use when re-loading the data using the memory loader function.

(5) L Memory loader function. This allows the user to load tapes which were saved using the print/punch function.

Several routines in the MIKbug monitor are written as subroutines and may be called by the user's program. The most frequently used of these subroutines are INEEE (input one ASCII character from the control port and put it in the A accumulator) and OUTEEE (output the character contained in the A accumulator to the control port). These subroutines start at \$E1AC and \$E1D1 respectively. Several modifications were made to these subroutines to allow the routing of all

output to ports other than the control port. These modifications are described in chapter V. Software listings are provided in Appendix A.

Summary

The M6800 microprocessor was selected primarily because of its treatment of I/O as memory, its low cost, and the availability of a software development system. An overview of the basic microprocessor system which includes a video terminal, a cassette interface, and a microprocessor has been presented to acquaint the reader with the microcomputer system which is used to control the testing of the NCR 2051. The reader is referred to References 5 and 7 for more detailed information concerning the M6800 microprocessor.

IV. DEVELOPMENT OF SYSTEM HARDWARE

Introduction

This chapter will describe the development of an interface that is required to connect the microcomputer to the memory under test (MUT). This interface must allow for either hardware or software control of the timing signals for write, erase, and read operations. The NCR 2051 has 16 data lines (bidirectional), 5 address lines (inputs), and 4 mode control lines (inputs). The interface must then allow for bidirectional data flow on 16 data lines and unidirectional data flow (from the processor to the MUT) on 5 address lines and 4 mode control lines. In addition, two power supply voltages are required (+5v and -29v). A parallel interface card is required for connection to the HP-9866A line printer, and a serial interface card is required to connect the system to the HP 9820A calculator system.

Parallel I/O Port Configuration

The parallel interface cards which are designed for the SWTPC M6800 microprocessor system will not provide the desired bidirectional data flow capability for the MNOS tester without some modifications. The MC6820 parallel interface adapter (PIA) integrated circuit is used on the parallel interface card. It consists of two eight bit

parallel bidirectional data ports. Each data port has one input control line and one bidirectional control line. (All bidirectional lines must be programmed for either input or output but may be reprogrammed under software control. Each line may be individually programmed for input or output.) One of these eight bit ports (called the B side) has TTL compatible output drivers which are capable of driving one TTL load. The other port (A side) has less drive capability but will drive a low power TTL load or a MOS circuit.

The parallel interface cards designate the B side of the PIA as all input lines. The A side is to be used as all outputs with CA1 and CB1 being input lines and CA2 and CB2 being outputs. All data lines (both input and output) are buffered so it is not possible to take advantage of the bidirectional capability of the PIA with a parallel interface card when configured the way the manufacturer suggests so some changes must be made in order to implement the required I/O lines.

The B side of the PIA is the better choice to use as a bidirectional port since it has enough drive capability to drive one standard TTL load (Ref 11) directly. This eliminates the need for the buffering and simplifies the interface design. Sixteen bidirectional data lines are required and the B side of two parallel interface cards will meet this requirement. In order to eliminate the buffering

on the B side and leave the A side buffers operational, headers are installed in the I.C. sockets with jumpers soldered between the appropriate pins to allow direct connections from the PIA B side to the I/O connector on the interface card. Since one I.C. on each card provided buffering for data lines on both the A and B sides of the PIA, it was necessary to install an I.C. socket on the header with connections made only to the pins which connect the buffers to the A side of the PIA (See Fig 4-1). With these changes, the two parallel ports provide the following I/O capability:

16 bidirectional TTL compatible data lines

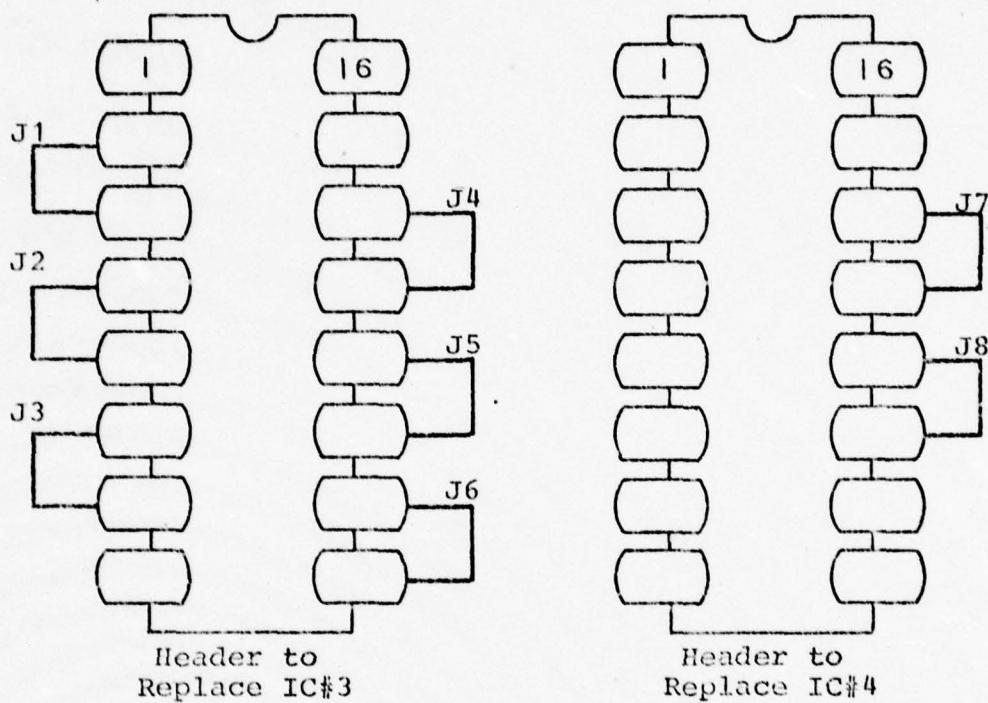
16 buffered output data lines

2 buffered output control lines

2 unbuffered input control lines

Interface to NCR 2051

Since all I/O signals to the NCR 2051 are TTL compatible, an interface is relatively simple to design. Only one pin on the NCR 2051 has a non-standard TTL voltage requirement and that is the -29v power supply. This input will accept any power which has a voltage of -29v plus or minus 1.5v. The current drain from this supply is most demanding when the memory is being read. The power supply must be capable of supplying a maximum of 12 ma (Ref 9). To meet this requirement, a dual power supply (+15v and -15v)



NOTE:

Before installing an IC socket on the header for IC#4, cut pins # 11, 12, 13, & 14 off the socket so that there is no contact between those pins on the IC and the corresponding pins on the header. There is no socket on the header which replaces IC#3.

Install jumpers J1 through J8 on the headers.

Fig. 4-1. Headers for MP-L Parallel Interface Cards

is selected. Two forward conducting diodes (1N 4004) are added to the output of the stacked power supplies (-30v) to reduce the output voltage to 28.8v (See Fig 4-2). A 28v power pak would have met this requirement more easily; however, it was not in stock and the lead time was too long to allow waiting for this particular supply. The 30 volt power pak (dual 15v) was in stock so it was used. The five volt power requirement is supplied by a three terminal voltage regulator (LM 340-T). The input voltage of eight volts is supplied to the voltage regulator by the unregulated eight volt power supply in the microprocessor (Ref 6).

Since the system being developed is to be used for laboratory testing of the memories as well as performing acceptance tests, it is desirable to be able to control the timing parameters and to provide a means of changing these parameters either with hardware (switches etc.) or with software. The finest resolution of timing which may be obtained with software changes is equivalent to the length of time required to execute the shortest instruction. For the M6800 the shortest instruction requires two machine cycles to execute. With the system clock of the SWTPC processor running at 898.5 khz (normal frequency), this corresponds to $1.113 \mu\text{s}/\text{cycle}$ or $2.226 \mu\text{s}$ as the finest resolution possible. Unfortunately the shortest time

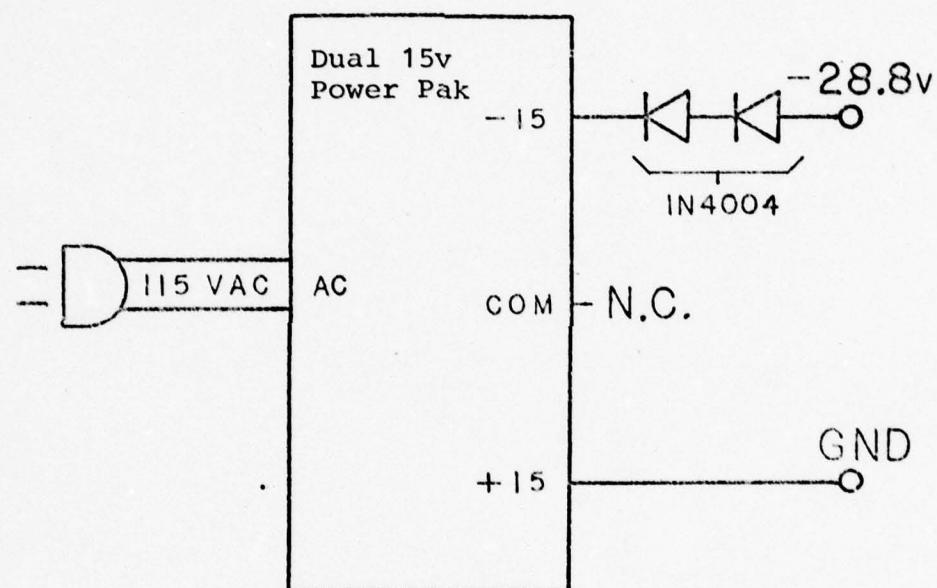


Fig. 4-2. Schematic Diagram for NUT Negative Power Supply

required to change the state of one output data line (generate a pulse) is the time required for the shortest instruction which will change the state of the output (a store accumulator instruction) to be executed (or five machine cycles). (The store accumulator direct instruction is executed in 4 cycles but cannot be used since the address of the I/O ports is not in the lower 256 bytes of memory). This corresponds to a minimum software I/O pulse time of 5.565 μ s.

While the processor minimum I/O pulse time and resolution time combination is acceptable for some timing requirements such as erase and write times (40 to 200 ms), it is not within the range of times needed for such things as the clock signal (2 to 20 μ s). Since software control of the erase and write times is possible using only the microprocessor hardware, it is accomplished with software and is described in chapter V. Two alternatives (to software) were considered for control of signals with times too short to be directly implemented with software: (1) hardware timing circuits such as one shot multivibrators and (2) hardware timing circuits which are software controlled such as presetable countdown timers. Due to the laboratory testing use of this system, the hardware circuit was selected since it gives greater resolution (continuous vs. incremental). If the system is to be used exclusively for

acceptance testing, the latter of the two methods is recommended since it provides for full software control over all timing circuits.

The address lines as well as the mode control lines are all input-only lines to the MNOS memory and therefore are straight-forward to interface with the PIA. In fact, the buffered outputs of the parallel interface cards are connected directly to the test socket for the memory under test (MUT). This includes all address and mode control lines except for the mode control line C1 (See Fig 4-3), where a TTL inverter is placed between the buffer and the C1 pin of the MUT test socket. This inverter is used so that a zero written into the PIA data register bit associated with C1 places the mode logic in a read mode rather than a write mode.

It is desirable from a laboratory testing point of view to be able to sample the data output lines of the NCR 2051 at some specified time relative to the read-clock pulse in order to evaluate the access time of the memory. This capability is provided by latches (See Fig 4-3) which are connected to the data lines of the MUT test socket and are controlled by one of the output lines from the parallel ports. The limiting factor with this method of control is also the processor speed. The shortest access time which may be tested using this method is 5.56 μ s with a 2.22 μ s

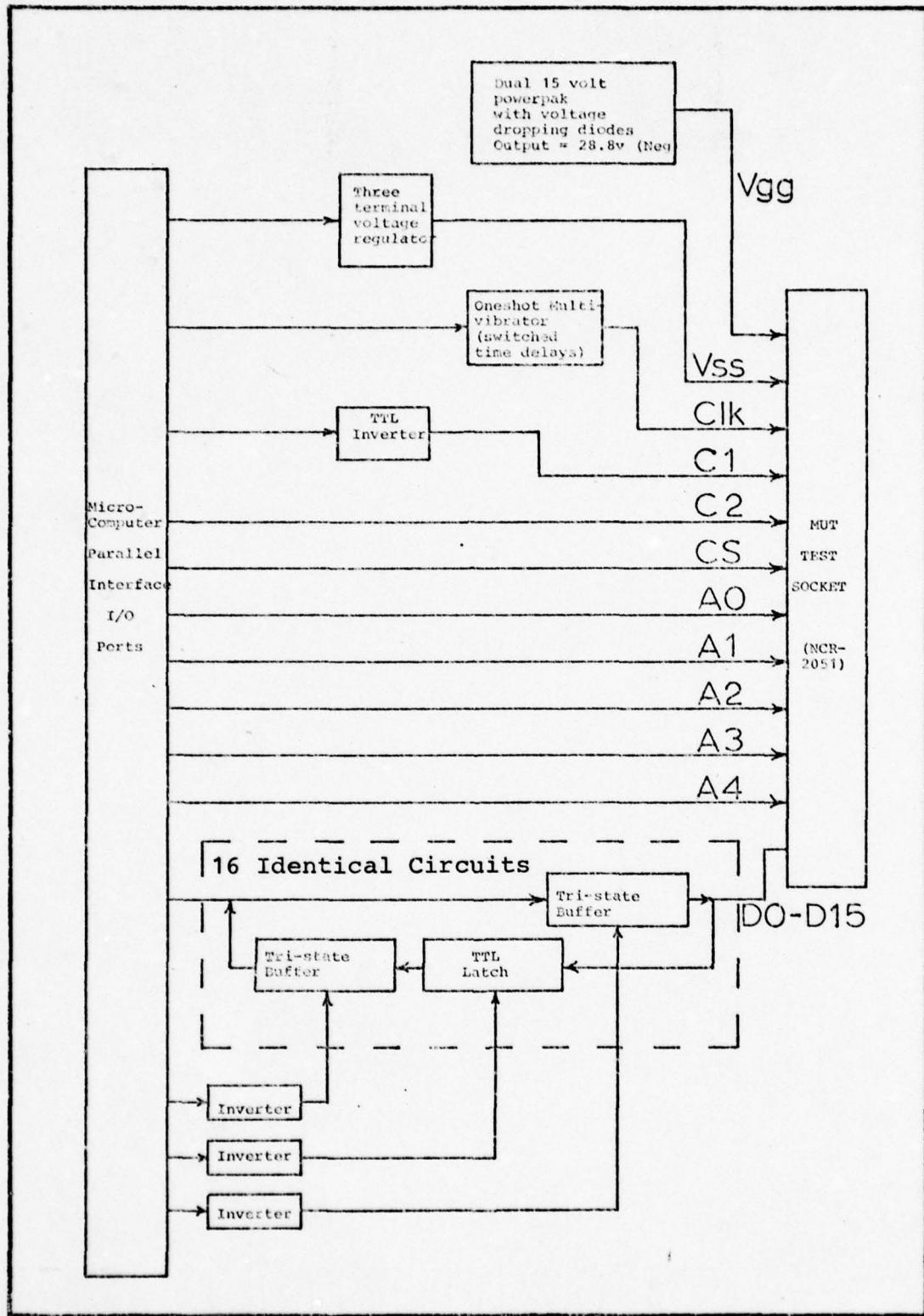


Fig. 4-3. Block Diagram of MUT Interface

resolution. If greater resolution or shorter test times are required, then the same choice as that made for the read clock timing circuit must be made. However, since in the applications for this device the speed is not a critical factor, the resolution thus obtained is satisfactory because we are not interested in evaluating the actual access time. We are instead only concerned with whether the access time is less than or equal to 5.56 μ s.

Since the PIA outputs have relatively low drive capability and the MUT is connected to the PIA with a four foot ribbon cable, it is necessary to provide buffering between the PIA and the MUT. This buffering will ensure that sufficient drive is provided to the MUT for proper operation. In order to provide buffering between the PIA and the MUT in both directions and to implement the above mentioned latching circuits, the circuit design shown in Fig 4-4 is designed to interface all sixteen data lines from the PIA to the MUT. The status of the tri-state buffers as well as the latches is controlled by TTL inverters which are driven by the buffered outputs from the PIA. The inverters are used so that the lines are active when ones are written into the PIA data register bit which corresponds to the line being considered.

An interface which connects the microcomputer to the NCR 2051 has been described. The interface includes TTL

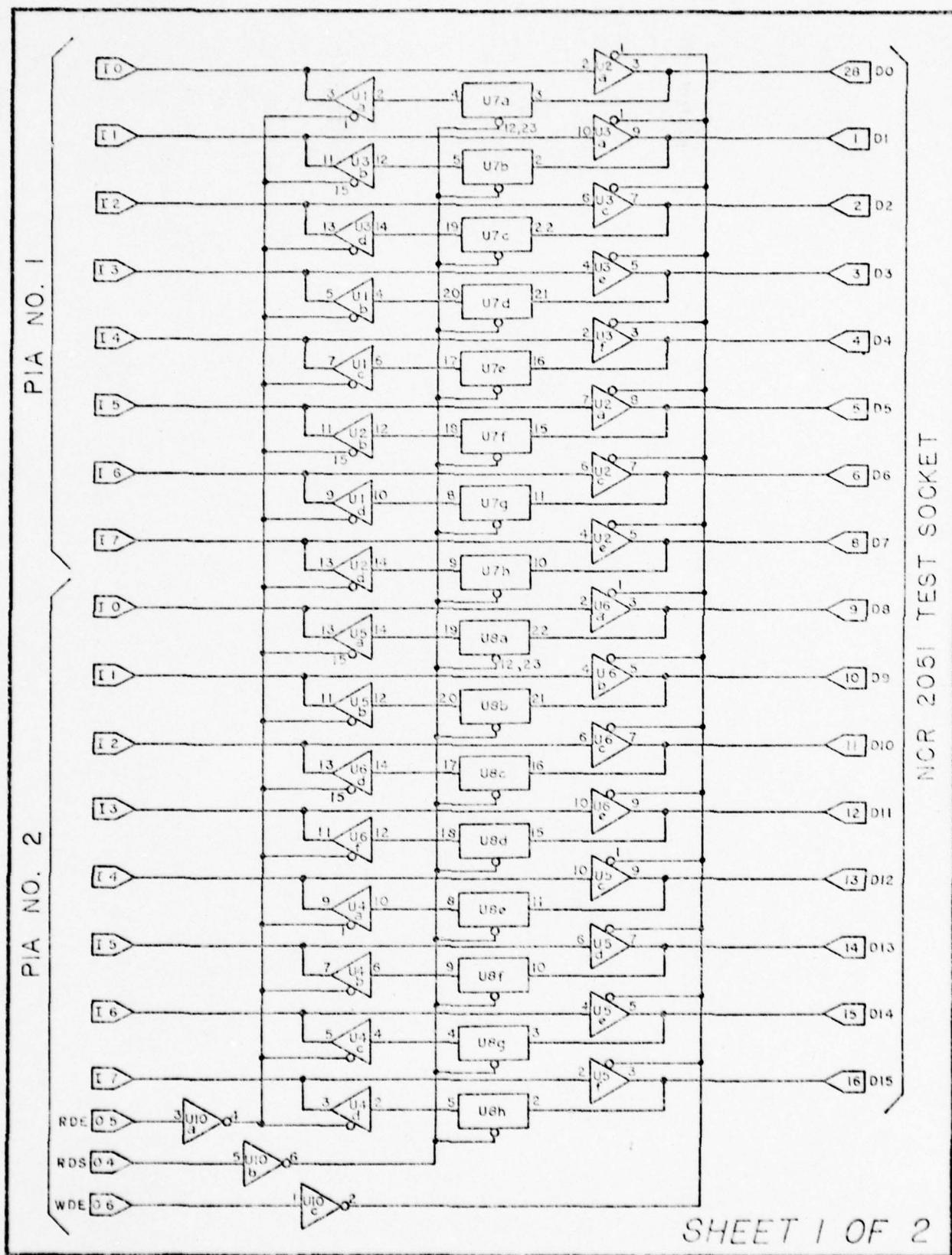


Fig. 4-4a. Schematic Diagram of MUT Interface

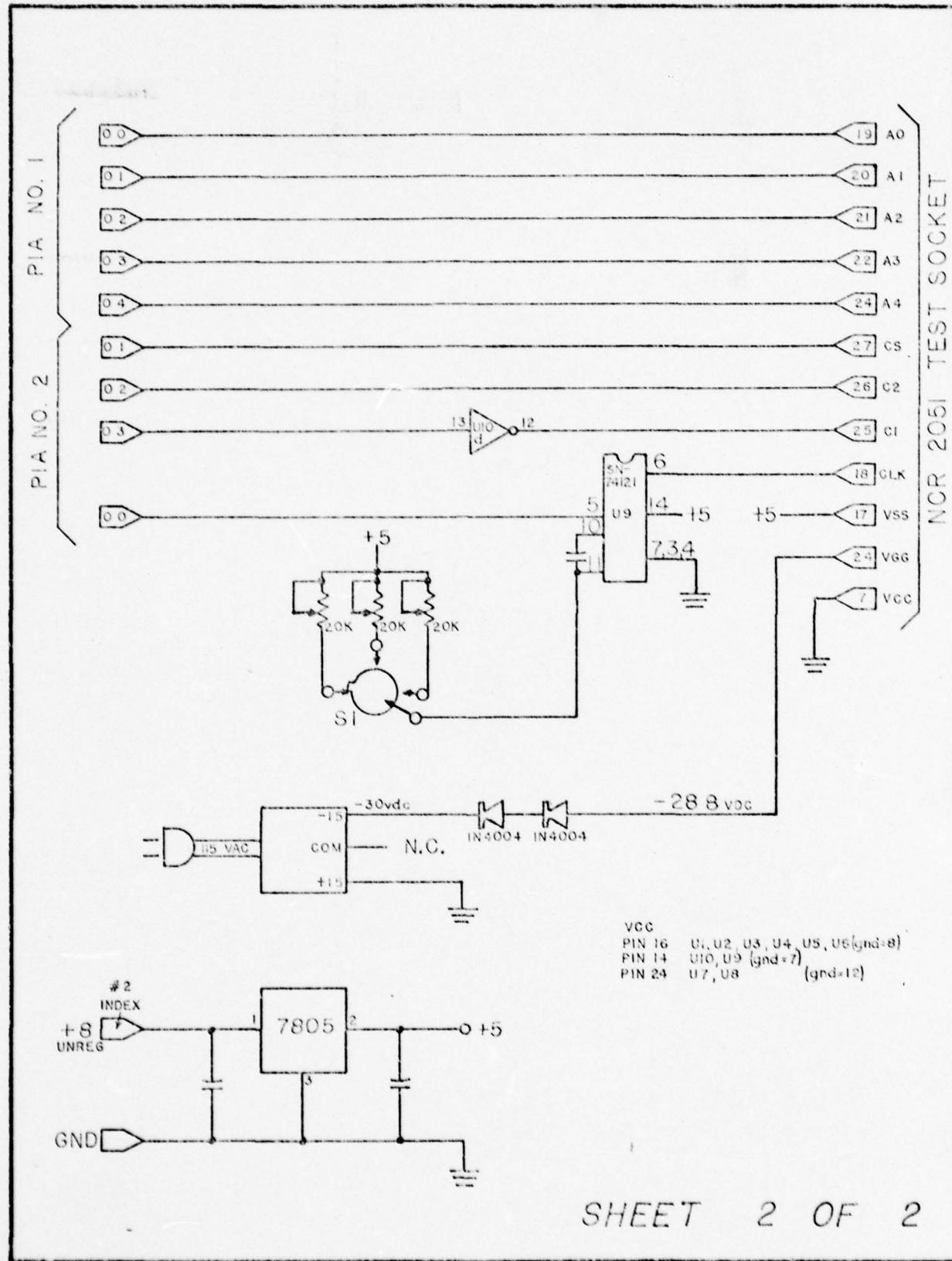


Fig. 4-4b. Schematic Diagram of MUT Interface
39 (b)

buffers, inverters, latches, a one shot multivibrator, and two power supplies (+5v and -28.8v). Software control is provided for erase and write pulse widths while hardware control is provided for the read-clock pulse.

Line Printer Port

In order to provide the capability to route data output to a line printer, an interface must be designed and constructed which will connect the test system to the line printer. This is accomplished by utilizing a parallel interface card provided by SWTFC. The buffered output side (seven bits from side A) of one parallel interface card can be connected directly to the seven bit ASCII input of the HP 9866A line printer through the connector provided on the back of the printer (Ref 3). Handshaking is provided through control lines CA1 and CA2 which may also be connected directly from the processor interface card to the line printer connector (See Fig 4-5). The Flag output from the line printer is an open collector device which requires a pull-up resistor. This resistor may be added to the parallel interface card and is to be installed between the +5v power supply and the CA1 input from the edge connector on the card. If the clear line from the line printer (clears printer line buffer) is to be used, care must be taken to shield it from the other lines to prevent crosstalk. Crosstalk can cause intermittent clearing of the printer

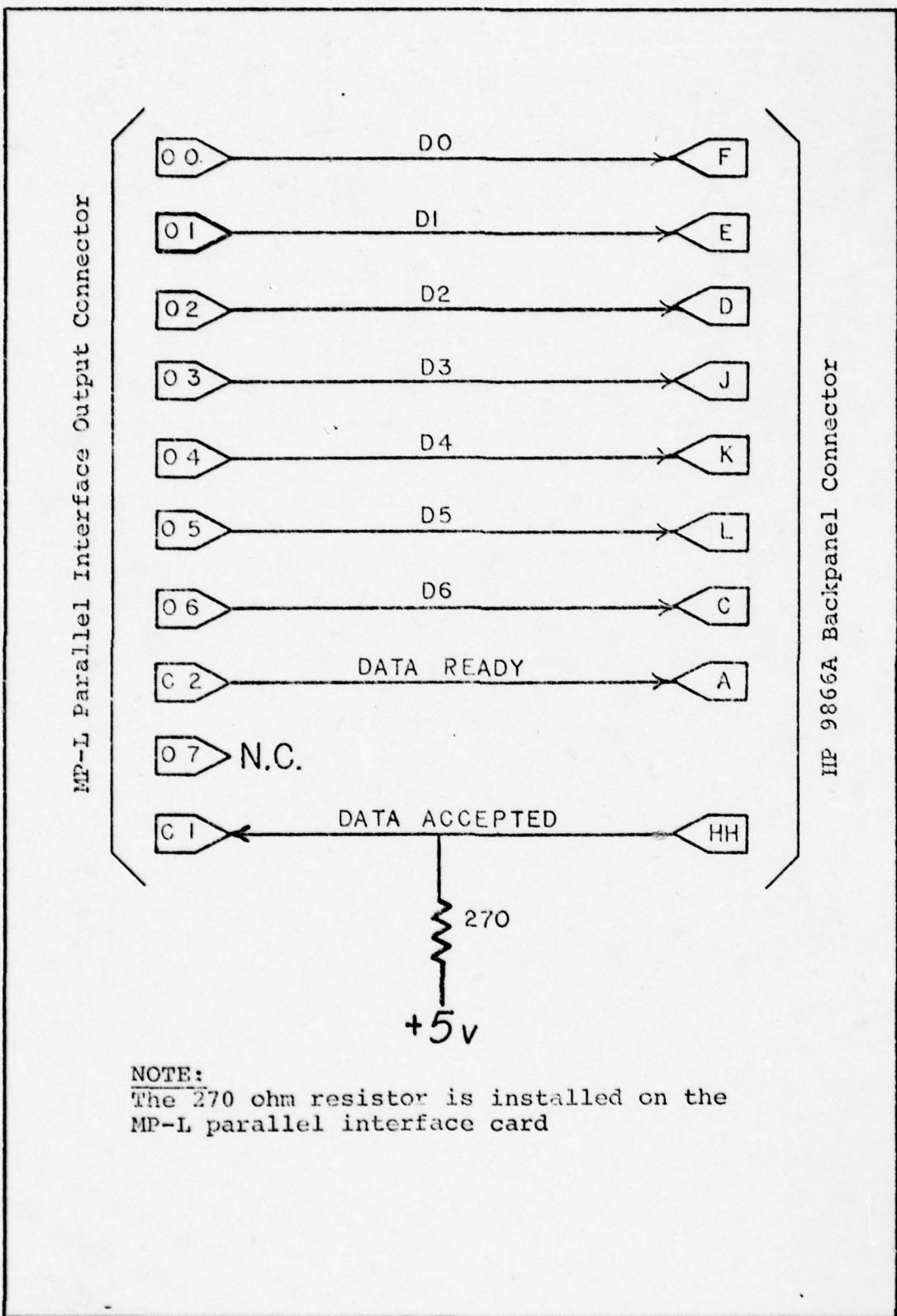


Fig. 4-5. Schematic Diagram of Line Printer Interface

line buffer and loss of information to be printed. It should be noted that the data which is written into the PIA data register for the line printer output must be complemented before it is written since the printer uses inverted logic at its inputs.

Serial I/O Port

The serial I/O port is to be used in two different output configurations (See Fig 4-6). Either a teletype using a 20 ma current loop at 110 baud or the HP 9820A calculator system using an RS-232 interface at 1200 baud may be connected to the serial port (Ref 6). The connection to the calculator system is made through a serial interface (HP model 11205A) which is provided by Hewlett Packard. This card plugs into one of the system party line buss slots in the HP 9820A controller and is user programmable for 110 to 1200 baud operation with 1 or 2 stop bits. For this application, operation at 1200 baud with 2 stop bits will be selected for rapid data transfer and programming compatibility with the 2 bit operation of the teletype (the serial port will always be programmed for 2 stop bit operation).

Microprocessor Board Modification

In order to make software changes to the MIRbug monitor program which is provided in factory programmed ROM (See

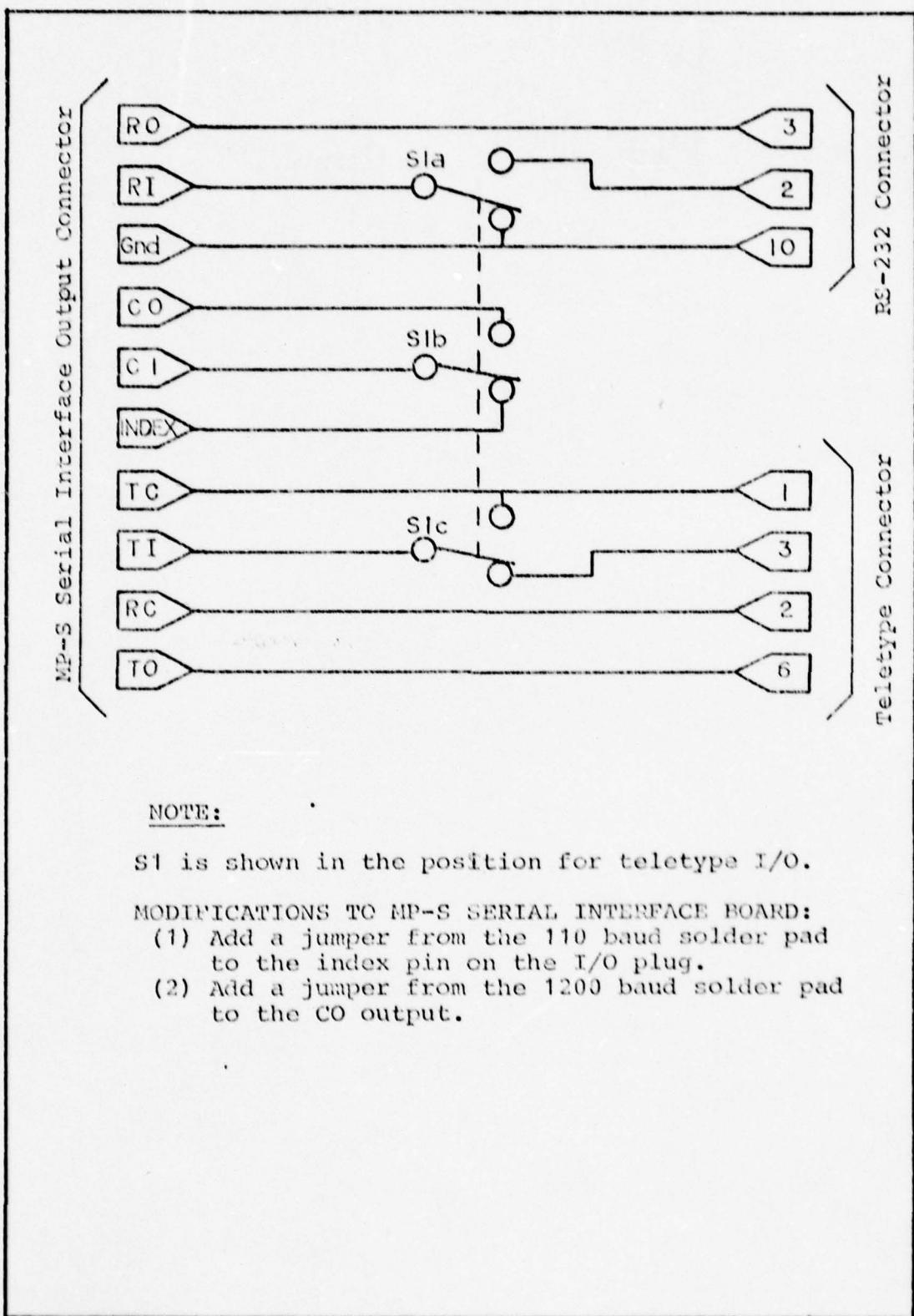


Fig. 4-6. Wiring Diagram of Serial I/O Port

System Monitor Chapter V.), it is necessary to change the circuit board which contains the ROM so that it will accept an EPROM which can be programmed with the modified version of MIKbug. To accomplish this, a circuit board was fabricated which contains a socket for a TMS 2708 EPROM, an SN7420, and a zener diode to regulate the -5v required by the 2708. Some resistors and capacitors were also included on the board. A header was installed so that the resulting assembled circuit board could be plugged into the existing socket for the MIKbug ROM. One conductor on the processor board must be cut and a jumper installed so that address line A9 runs to pin 15 on the MIKbug ROM socket. This additional address line is required since the 2708 occupies 1024 bytes of memory space while MIKbug occupies only 512 bytes. Two jumpers must also be installed from the +12v and -12v pins on the processor board edge connector to the new circuit board which is plugged into the MIKbug ROM socket (See Fig 4-7).

Summary

The hardware design for connection of the microprocessor system to the MUT has been discussed. The modifications to the SWTPC parallel interface cards which are necessary for their use to drive the interface are described in detail. The interface itself, along with timing considerations, is discussed. A line printer to

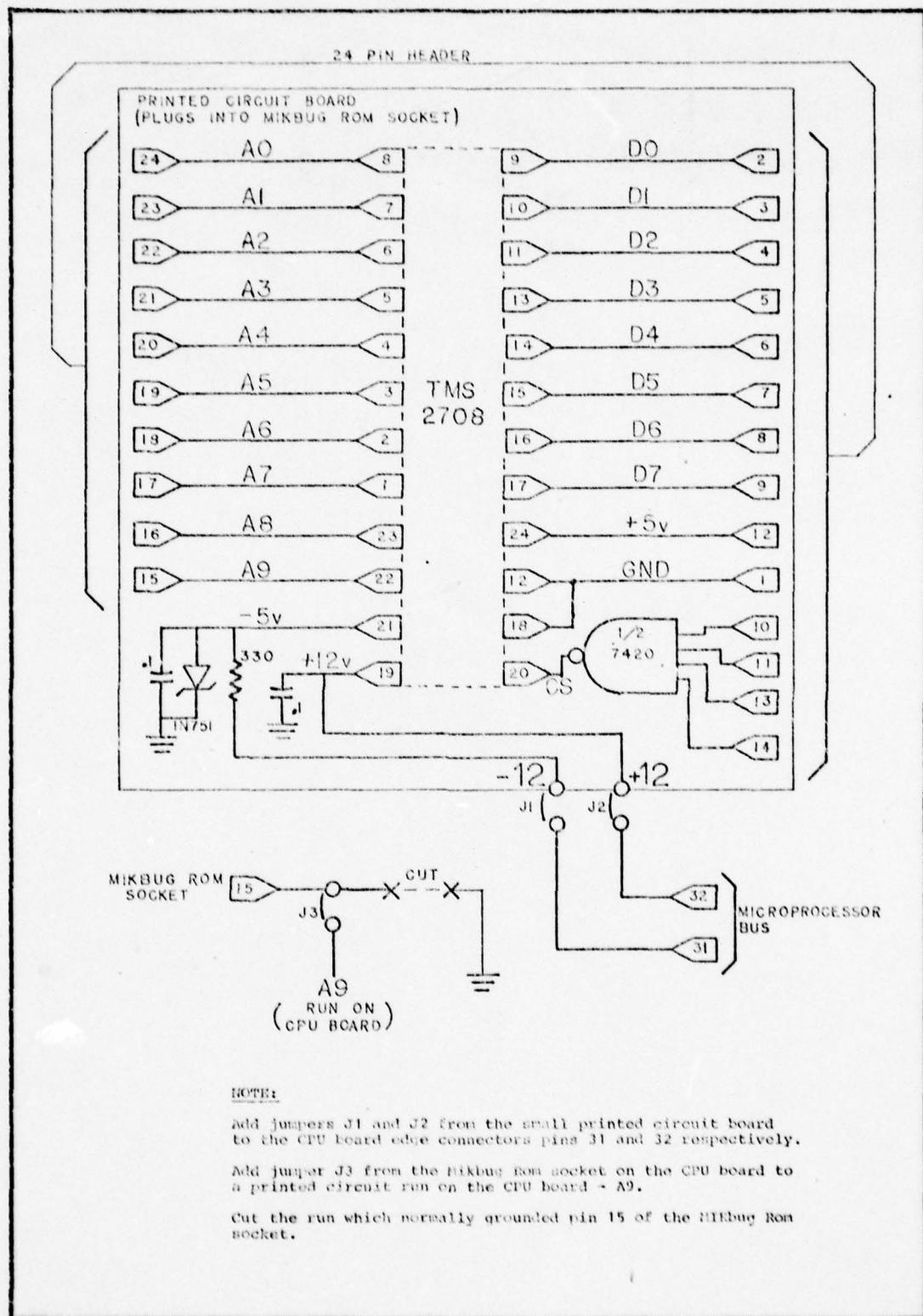


Fig. 4-7. Schematic Diagram of CPU Board Modifications

microprocessor interface, as well as a calculator system/teletype to microprocessor interface, have also been described.

V. SYSTEM SOFTWARE DEVELOPMENT

Introduction

This chapter describes the development of the software which is necessary to implement the testing of the NCR 2051. The software must sequence the microcomputer system in such a way as to implement the test algorithms which have been developed based on the acceptance testing requirements. These algorithms are described in detail here, (See Appendix B for software listings). In order to simplify the transfer of information to the desired output device (control terminal, HP 9820A calculator system, HP 9866A line printer, or a teletype) a system monitor is developed based on a modified version of the MIKbug monitor and is presented in this chapter. A monitor for the testing routines is needed in order to allow the operator to select the desired test algorithm and control the related timing parameters. A discussion of this monitor followed by a description of some of the frequently called general subroutines is also contained in this chapter.

Algorithms

Since pattern sensitivity tests are the most effective method of testing the MNOS memory (Ref. 10), four different algorithms which are commonly employed to test random access memories (RAM) have been implemented for the NCR 2051 using the M6800 microcomputer system. The software has been developed using subroutines extensively. This allows the addition of future software without having to duplicate the existing functions which are used in the new software. Flowcharts of these algorithms are included in Appendix E. A description of each algorithm follows:

MARCH (Write and read/write forward and backward)

A background pattern is written into all memory locations. All addresses are then read and compared with the background pattern to verify that the entire memory contains the background pattern. If any errors are detected, the type of the error is printed on the output device (ERR-BCKGND • VERIFY) followed by the data which is written, the data which is read, and the address at which the error occurs. Each occurrence of an error causes an error message to be printed.

Starting with the lowest address, each memory location is read and again compared with the background test pattern. If an error is detected, the type of error is printed on the output device (ERR-TYPE 1) followed by the written data, the

read data, and the address of the error. The memory location is then written with a test word (usually the complement of the background pattern, since this represents the most rigorous test, but it is selected by the system operator). This sequence is continued (incrementing through memory) until the top address is reached. At this time, the entire memory contains the test word.

Starting with the top address this time, each memory location is read and compared with the test word. If an error is detected, the type of error (ERR-TYPE 2) is printed on the output device followed by the data written (test word), the data read, and the address of the error. The background pattern is then re-written into that memory location and the sequence continues (decrementing) until the bottom address of memory is reached. At this time the memory is once again filled with the background pattern. The background pattern is again verified starting with address zero and incrementing through memory. Errors are noted by printing ERR-BCKGND VERIFY followed by the written data (background pattern), read data, and the address of the error. Since all of the test programs are written as subroutines, a return from subroutine instruction is then executed.

MASEST (Alternating 1's and 0's)

Starting with location zero, alternating words of all

ones or all zeroes are written throughout the memory. This provides an effective test for interaction between adjacent memory cells since they will have the alternating pattern stored in adjacent cells. (This program is automatically executed twice with all ones written into location zero the first time and all zeroes written into location zero the second time.) The memory is sequentially read (starting with address zero) and the contents of each memory location verified. If an error is detected, ERR-TYPE 6 will be printed on the output device followed by the data written, data read, and the address of the error. The memory is then read and the contents verified in the following sequence starting with address zero and ascending through the memory: address, complement of address, address; address + 1, complement of address + 1, address + 1, etc. If an error is detected, ERR-TYPE 4 will be printed on the output device for odd addresses (ERR-TYPE 5 for even) followed by the data written, data read, and the address of the error. Again starting with address zero, the memory is sequentially read and the contents of each location verified. As with all of the test routines, when the second run is completed a return from subroutine is executed.

WAKPAT (Walking pattern)

A background pattern is written throughout the memory and then verified. Errors detected are reported as

ERR-BCKGND VERIFY followed by the data written, data read, and the address of the error. Starting with address zero, a test word is written into one location then the entire memory is sequentially read and verified (starting with address zero and incrementing). The background pattern is then restored to the address which contains the test word and the next higher address is written with the test word etc. until the memory has been fully tested using this sequence. Errors which are detected are reported as ERR-TYPE 1 for locations which should contain the background pattern (ERR-TYPE 2 for those which should contain the test word) followed by the usual string of data. When execution is complete, the MUT will contain the background pattern in all locations and a return from subroutine instruction is executed.

GALPAT (Galloping pattern)

A background pattern is written throughout memory and then verified. Errors are reported as ERR-BCKGND VERIFY followed by the data written (background pattern), the data read, and the address of the error. A test word is written into one location starting with address zero, then the memory is read and each location verified in the following sequence: first background location, test word location, first background location; second background location, test word location, second background location; etc. Errors

detected are reported as ERR-TYPE 7 for background locations (ERR-TYPE 8 for test word locations) followed by the data written, the data read, and the address of the error.

The background pattern is then restored to the test word location and the test word written into the next higher address. The verify routine is then repeated. This sequence continues until the test word reaches the top address in memory. The contents of the test word and the background pattern are then swapped and the entire test procedure is repeated. After the second pass, the contents of the background pattern and the test word are again swapped so that they will be the same as when the test program was entered. The MUT will contain the original test word in all memory locations when the return from subroutine instruction is executed.

System Monitor

A microcomputer system should have a system monitor which takes care of certain housekeeping chores such as initially programming the I/O port through which the operator communicates with the system. An alternative to this approach is a switch panel which is designed to allow the operator to read and write data from memory locations by depressing switches and observing some sort of optical display to determine the state of each bit in a memory location. The switch panel is not very practical for the M6800; however, and it is not available on any M6800 based microcomputer kit (this is due to circuit complexity and cost). The software is to be written so that all input/output communications between the microprocessor and the human operator pass through the system monitor (i.e. modified version of MIKbug). Rather than write several I/O routines, it is more efficient to modify the MIKbug monitor to provide for routing the output data to one of three I/O ports. With this method, the operator may route any output which would normally appear on the control terminal to a parallel port (connected to the HP line printer) or to a serial port (which may be connected to either a teletype or to the HP calculator system).

Since the MIKbug monitor is provided in ROM, it is necessary to perform some hardware modifications to the main

processor board of the system in order to install an EPROM containing the new version of the system monitor. This modification is described in chapter IV. A TMS 2708 EPROM is used to provide space for a one kbyte monitor to replace the 512 byte MIKbug monitor.

The input and output character routines are changed so that if the route command is enabled (Flag1 = 0) a specified input character causes the route selection program to be entered (See flowcharts in Appendix E.). A control P is selected because it is a character which is not used for any other purpose within the system. Upon entry into the routine, the comment "SELECT DESIRED OUTPUT DEVICE - TER, TTY, PRTR" is printed on the control terminal (independent of the previous output destination). The user then responds with at least the first two characters of the output device code (TE, TT, or PR) followed by a carriage return. A message is then printed to assure the user of the output destination and Flag2 is set to direct the system output to the proper output device (if Flag2 = 0 then the output goes to the control terminal, if Flag2 is positive then the output goes to the serial port, and if Flag2 = negative the output goes to the line printer). Control is then returned to whatever program was running when the route command was entered. The route selection program is completely transparent to any program that may be in progress.

It may be desirable in some instances to inhibit the route selection program (such as when loading cassette tapes, which have been recorded in binary format and may contain the equivalent of an ASCII control P, into the system). This is easily accomplished by setting Flag1 to any nonzero value before entering the loader program. As a precaution, the MIKbug memory loader function is changed to provide this feature even though a control P should not be encountered on a tape which is recorded in the MIKbug format. Due to the method used to end the MIKbug memory loader function, (the return is shared with the return from other MIKbug functions) it is necessary to add a third flag to indicate when a load is in progress. This directs the return routine to re-enable the route command if a load is in progress.

Both port initialization and output character routines are included to drive the two additional output ports. The data output to the serial port is in standard ASCII code, but the data output to the parallel port is complemented ASCII code as the input to the printer is inverted logic.

Test Monitor

Since it is desired that the operator control which test program is to be executed, a monitor program is necessary to allow the operator of the test system to select the desired test program. The names of each test subroutine and the codes which select the subroutines are displayed on the output device. The monitor also enables the operator to select the desired pulse widths (for write and erase) as well as the background test pattern and test word which is used by the test programs (See flowchart in Appendix E.). This enables the test programs to be executed using the minimum and maximum pulse widths listed in the specifications for the NCR 2051. (When the system is to be used in a laboratory testing application, it is desirable to have the capability to execute each test program with any operator specified pulse widths and data patterns for the background pattern and the test word.)

When the test monitor is entered, the names of all the subroutines and the code used to select each subroutine are displayed on the output device which is selected by the output selection routine. The display appears as in Fig 5-1. This gives the operator an immediate list of all the codes from which he may look up the code for the particular program he wishes to run. It also serves as a directory of all available test programs and may be lengthened as new

programs are added. In order to place the entire test software package in a 2 kbyte EPROM memory block, no headings are displayed at the top of each column. This should present no problems as the operator will soon memorize the meaning of each column and will have no need for the headings.

SELECT DESIRED PROGRAM

01 MARCH

02 MASEST

03 GALPAT

04 WAKPAT

0D DISPLAY

FF PULSE WIDTHS DISPLAY/CHANGE

?

Fig. 5-1. Test Program Directory

A program is selected by typing in a two digit code (leading zeroes are not necessary) followed by a carriage return. The input routine is written so that it accepts only the last two entries. If an error is made while typing the input code, it may be corrected by simply retying both digits (if only one digit is re-entered the program will look at the last two entries and produce an error) or, if desired, a control X will cause the deleted message to be printed and allow the input to be re-entered. If a carriage return is entered before the error is noted, a control C returns the system to the test monitor entry point and the test routine list is printed again. The control C has this effect at anytime except when the system is executing a test program (a control C will be ignored in this case).

The first four entries in the test subroutine directory are test programs which run a specified test on the MUT and are explained in detail later in this chapter. The fifth entry is the subroutine that displays the contents of the MUT. This program prints each address followed by its contents so that the complete memory is displayed on a 16 line CRT display at one time.

The sixth entry is the pulse widths display/change subroutine. This program displays the contents of a pair of two byte temporary storage locations which represent the current pulse widths associated with the write and erase functions. These concatenated memory bytes are loaded into the index register to provide a software delay loop whose length is determined by the number contained in the memory locations (the number of times the delay routine passes through the delay loop is equal to the number contained in the memory storage locations). When the test monitor is entered, these two registers are preset to 1000 HEX (all entries are intrepreted as hexadeciml numbers). When the display/change routine is entered, the current value stored in the two registers will be displayed and the user may input new values if so desired. Once a change is made, the new write/erase times will be displayed and will remain in effect until they are changed by using the display/change routine or by entering the test monitor through the initial

entry point (values are then reset to 1000 HEX). The times which the hexadecimal numbers represent may be found in Appendix D.

General Subroutines

Since subroutines are used extensively in the system software development, it is important to describe the operation of some of the most frequently used routines in detail. (See Appendix B for complete software source listings.)

WINIT (Write initialization)

This subroutine performs the required initialization of the PIA registers to program the I/O ports for a write operation. It is entered only if the previous I/O operation to the memory under test (MUT) is a read operation.

WRITE

The write subroutine causes the data stored in WDATA to be written into the MUT at the address that is stored in ADDR. (See Flowchart in Appendix E.) Upon entry into this subroutine, it is assumed that the address, data to be written, erase pulse width, and write pulse width have all been stored in the appropriate memory storage registers. (The address is stored in the memory register called ADDR, the data to be written is stored in WDATA, the erase pulse width is stored in EPWID, and the write pulse width is stored in WPWID.) The routine first tests to see if the I/O ports are programmed for a read or for a write operation. If they are programmed for reading then a branch to subroutine

is issued to the WINIT subroutine to program the ports for writing. The WMODE flag is then tested to determine what operation(s) is(are) to be performed on the MUT. If WMODE = 0, an erase operation is performed followed by a write operation. If WMODE is positive, only a write is executed. An erase only mode is selected if WMODE is negative. The pulse widths for the erase and write functions are determined by the contents of memory storage registers EPWID and WPWID respectively.

RINIT (Read initialization)

If the previous I/O operation was a write operation, (if the contents of STATUS is non-zero then RINIT is called) this subroutine is called by the Read subroutine to program the I/O ports for a read operation.

READ

This subroutine first tests to see if a read initialization is necessary (STATUS is non-zero) and, if so, issues a branch to the subroutine RINIT. (See flowchart in Appendix E.) It then reads the MUT at the address contained in the memory register ADDR and places the read data in memory register RDATA. Since the read clock signal pulse width is determined by hardware, there are no software delay loops associated with this subroutine, as there are in the WRITE subroutine.

INHEX

The INHEX routine is used to input a hexadecimal character string. It retains only the last four inputs, so corrections are made by simply re-entering the data or by typing a control X (this causes the deleted message to be printed on the output device). The routine is used to input two characters as well as four characters and is terminated with a carriage return. A control C input causes a jump to the test monitor re-entry point while an X input causes a jump to the MIKbug monitor.

Summary

Four algorithms which perform pattern sensitivity tests on the NC 2051 have been described. These algorithms (MARCH, MASEST, WAKPAT, and GALPAT) use four different sequences to write data into memory, then read and verify that the proper data was stored and/or read. Executing these four algorithms tests all memory cells within the memory as well as the addressing and logic sensing elements of the memory. A system monitor allows the operator to control the output destination of all output messages. The test monitor is used by the operator to preset the test variables and select the desired test algorithm or subroutine.

VI. RESULTS, CONCLUSIONS, AND RECOMMENDATIONS

The objective of this effort was to develop a microprocessor based test system for the NCR 2051. This test system has a twofold purpose: (1) use as a model for an acceptance testing system and (2) use in a laboratory for laboratory-type testing of the NCR 2051. This objective has been realized as all the necessary design, development, and implementation of the acceptance testing system has been accomplished. The system's testing capability can be easily expanded through the addition of software, utilizing existing subroutines. As required, a user's manual is provided (Appendix C) and the test monitor displays prompting messages. This allows operation of the system without referring to the manual except to obtain an unfamiliar pulse width value.

Test Results

In order to verify the proper testing sequence, the algorithms were executed without a memory installed in the MUT test socket. This causes an error at all memory locations if the background pattern and the test word contain data other than all ones. (The background pattern and test word were preset to a value other than all ones before the algorithms were executed.) The error printouts then provide an effective trace function of the program

since each attempt at verifying the data contained within a memory location will result in an error statement printout. The software was written using subroutines extensively and therefore will allow easy modification to accomodate many different types of tests which might be desirable to perform in the laboratory.

Several NCR 2051 devices were tested using this test system. They were all found to be acceptable devices when tested at room temperature. Some devices which were known to be defective were also tested and the test system indicated that they were bad by the error statements which resulted. The author had planned to perform testing of several devices at elevated temperatures, but the necessary hardware did not arrive in sufficient time to fabricate an extension cable to allow the MUT to be placed into a temperature chamber.

Hardware

The system includes the following items which are necessary to accomplish acceptance-type tests: a microprocessor system (equipped with a minimum of three kbytes of programmable read only memory and two modified parallel interface cards plus one control interface card), a control terminal (a video terminal is used but could be replaced with a teletype if a printout is desired), and an interface to the NCR 2051 MUT. Additional items which are desirable for laboratory use include: 16 kbytes of read and

write memory (RAM), a cassette tape interface, a Hewlett Packard 9866A line printer, and a Hewlett Packard 9820A calculator system with its associated peripherals (papertape punch, papertape reader, X-Y plotter, and digital cassette tape memory system).

Software

The software package as developed for this system provides four test routines which perform pattern sensitivity tests and may be used for acceptance testing. These four test routines are called MARCH, MASEST, GALPAT, AND WAKPAT. The erase/write pulse widths are software controlled and the read-clock pulse width is hardware controlled. A routine is included which displays the contents of the MUT. The read/write/erase routines are written to allow the user to easily expand the software package with new routines if desired for laboratory testing.

Conclusions

This test system can be used to perform acceptance testing of the NCR 2051 with a high degree of confidence that defective memories will be identified. The system also has great potential for laboratory testing and, with the addition of the necessary software, can perform virtually any desired test on the NCR 2051.

Recommendations

This system will be very useful for laboratory testing; however, since the primary use of this system is for acceptance testing, the software which has been developed does not provide all of the desired test routines for this application. One important test, which could be added to the software, writes both ones and zeroes into the same location (without erasing) with different ratios of write times (See Ref 10). This procedure stores charge on both MNOS transistors associated with each memory bit cell and important retention data can be extrapolated with this method. The erase/write subroutines have been written in such a manner as to make this particular test easy to add to the existing system.

The techniques used for testing the NCR 2051 can also be applied to testing TTL devices. Since the test algorithms were selected to perform the most efficient tests (due to the device degradation effect when write/erase cycling the NCR 2051), the use of these algorithms should reduce the time required to accomplish TTL memory testing when compared to testing every possible bit combination as is frequently done.

Bibliography

1. Engineering Note 100. Phoenix, Arizona: Motorola Incorporated, 1975.
2. HP 9820A Calculator Operating and Programming. Mountain View, California: Hewlett Packard Incorporated, 1972.
3. HP 9866A/B Printer Service Manual. Mountain View, California: Hewlett Packard Incorporated, 1976.
4. MD-104 LSI Test System, Technical Specifications. Woodland Hills, California: Macrodata Corporation, undated.
5. M6800 Microprocessor Applications Manual (First Edition). Phoenix Arizona: Motorola Incorporated, 1975.
6. M6800 Microprocessor Operating Manual. San Antonio, Texas: Southwest Technical Products Corporation, undated.
7. M6800 Microprocessor Programming Manual (Second Edition). Phoenix, Arizona: Motorola Incorporated, 1975.
8. Peripheral Interface Adapter (PIA) Data Sheet. Phoenix, Arizona: Motorola Incorporated, undated.
9. Preliminary Data - NCR-2051. Miamisburg, Ohio: National Cash Register Corporation, 1970.
10. Schuermeyer, Fritz. "Test Results on an MNOS Memory Array." IEEE Transactions on Electron Devices Vol. ED-24 No.5 (May 1977): 564-568.
11. The TTL Data Book for Design Engineers (Second Edition). Dallas, Texas: Texas Instruments Incorporated, 1976.

Appendix A

System Monitor Software Listing

Appendix A includes the M6800 assembly language source listing for all modifications made to the MIKbug operating system. These modifications are discussed in Chapter V in the section entitled "System Monitor".

00010 NAM MONITOR FOR WAROM SYSTEM

00030 *****
00040 *****
00050 *****
00060 *** ***
00070 *** WRITTEN BY ***
00080 *** JOEL W. ROBERTSON ***
00090 *** ***
00100 *** 17 FEBRUARY 1978 ***
00110 *** ***
00120 ***
00130 ***
00140 ***

00160 * THIS IS A PATCH FOR 'MIKBUG' AND SHOULD BE
00170 * LOADED OVER MIKBUG (OVERLAY) SO THAT ALL LOCATION
00180 * LISTED ARE CHANGED TO THE INDICATED VALUES
00190 * (ALL OTHER MIKBUG LOCATIONS REMAIN THE SAME AS
00200 * IN THE ORIGINAL MIKBUG). IT IS INTENDED THAT
00210 * THE MIKBUG ROM BE REPLACED WITH A TMS-2703 TO
00220 * ALLOW THE ADDITIONAL ROUTINES AND CHANGES.
00230 * THIS EPROM ALLOWS AN ADDITIONAL 512 BYTES OF ROM.

00250 * THIS PROGRAM MODIFIES THE INPUT AND OUTPUT
00260 * ROUTINES USED BY MIKBUG TO ALLOW THE USER
00270 * TO SELECT ONE OF THREE OUTPUT DEVICES TO
00280 * BE USED FOR ALL SYSTEM OUTPUT WHICH UTILIZES
00290 * OUTEE AND INEE IN MIKBUG.

00310 * THE OUTPUT ROUTING PROGRAM IS ENTERED BY
00320 * TYPING A CONTROL P AT THE INPUT TERMINAL.
00330 * THE ROUTING PROGRAM WILL BE ENTERED ONLY
00340 * IF MEMORY LOCATION \$A014 IS EQUAL TO ZERO.
00350 * (THIS MEMORY IS SET TO ZERO BY THE POWER UP
00360 * ROUTINE BUT MAY BE CHANGED BY THE USER TO
00370 * ALLOW BINARY TAPES TO BE LOADED INTO THE SYSTEM.)

00390 * THE USER MAY WRITE PROGRAMS WHICH CHANGE
00400 * THE OUTPUT DEVICE WITHIN THE PROGRAM AS
00410 * LONG AS THE PIA (OR ACIA) IS INITIALIZED.
00420 * FLAG2 = 0 FOR TERMINAL
00430 * FLAG2 = POS. FOR TTY
00440 * FLAG2 = NEG. FOR LINE PRTR.
00450 * THIS ALLOWS THE OUTPUT OF DATA ONLY THROUGH
00460 * ONE DEVICE AND MESSAGES MAY BE SENT THROUGH
00470 * ANOTHER DEVICE WITHOUT INTERRUPTING THE
00480 * SEQUENCE OF DATA.

PAGE 002 MONITOR

00500 * THE TTY ACIA MAY BE CONNECTED TO THE HEWLETT-
00510 * PACKARD 9820 CALCULATOR SYSTEM INSTEAD OF THE
00520 * TELETYPE FOR DATA TRANSFER INTO THE CALCULATOR
00530 * SYSTEM THROUGH THE SERIAL INTERFACE CARD.

```

00550          * DEFINE PERTINENT AREAS OUTSIDE PROGRAM
00560      A014  EQU     $A014
00570      A015  EQU     $A015
00580      A016  EQU     $A016
00590      A012  XTEHP  EQU     $A012
00600      E07E  PDATA1 EQU     $E07E
00610      E1A5  SAV     EQU     $E1A5
00620      E1AF  IN1    EQU     $E1AF
00630      E1D4  IOUT   EQU     $E1D4

00650          OPT     O,NOGEN,S

00670  E00C          ORG     $E00C
00680  E00C  BD E398  JSR     LOAD    GO DISABLE ROUTE COMMAND
00700  E044          ORG     $E044
00710  E044  7E E3A2  JMP     LODEND  GO ENABLE ROUTE COMMAND
00730  E0D6          ORG     $E0D6
00740  E0D6  7E E389  JMP     PWRUP   JMP TO MODIFIED RESET ROUTINE

00760          ****
00770          ****
00780          *
00790          * THE FOLLOWING IS TO BE USED ONLY
00800          * WHEN THE TEST PROGRAM HAS BEEN
00810          * INSTALLED IN EPROM -- THIS ALLOWS
00820          * THE TEST PROGRAM TO BE ENTERED USING
00830          * A 'T' COMMAND FROM MTBUG
00840          * (SEE LABEL 'COMAND' FOR ADDITIONAL INFO)
00850          *
00860  E10B          ORG     $E10B
00870  E10B  7E E3B0  JMP     COMAND
00880  E10E  01        NOP
00890          *
00900          ****
00910          ****

00930  E1AC          ORG     $E1AC
00940  E1AC  7E E200  JMP     ICHROU  JMP TO MODIFIED INPUT CHAR
00960  E1D1          ORG     $E1D1
00970  E1D1  7E E349  JMP     OCHROU  JMP TO MODIFIED OUTPUT CHAR

00990  E200          ORG     $E200
01000      E200      ICHROU EQU     *
01010  E200  7D A014  TST     FLAG1

```

```

01020          * IF FLAG1 NOT= 0, THEN IGNORE ROUTE COMMAND
01030 E203 27 05      BEQ    *+7
01040 E205 37      PSHI B
01050 E206 8D 9D      BSR    SAV
01060 E208 20 A5      BRA    IN1
01070          * RETURN TO NORMAL OUTEEE IN MIKBUG

01090 E20A BD E28E      JSR    INCH1
01100 E20D 81 10      CMP A  #$10      TST FOR CONTROL P
01110 E20F 27 01      BEQ    *+3
01120 E211 39      RTS
01130          * DO NOT ENTER ROUTE PROG IF INPUT
01140          * IS NOT A CONTROL P

01160 E212 PRTMES EQU  *
01170 E212 FF A012      STX    XTIMEP
01180 E215 86 00      LDA A  #0
01190 E217 B7 A015      STA A  FLAG2      PROG FOR TER OUTPUT
01200 E21A CE E2A5      LDX    #MESS1
01210 E21D BD E07E      JSR    PDATA1
01220 E220 FE A012      LDX    XTIMEP

01240 E223 8D 69      BSR    INCH1
01250 E225 81 54      CMP A  #'T
01260 E227 26 3F      BNE    CP

01280 E229 8D 63      BSR    INCH1
01290 E22B 81 45      CMP A  #'E
01300 E22D 26 19      BNE    CT

01320          * TERMINAL SELECTED
01330 E22F 8D 5D      WA1    BSR    INCH1
01340 E231 81 18      CMP A  #$18      TST FOR CTRL X
01350 E233 27 DD      BEQ    PRTMES      GO BACK TO START
01360 E235 81 0D      CMP A  #$D      TST FOR CAR RET
01370 E237 26 F6      BNE    WA1      WAIT FOR CAR RET
01380 E239 FF A012      STX    XTIMEP
01390 E23C 8D 60      BSR    PRINT      GO PRNT 1ST PART OF MESSAGE
01400 E23E CE E2D8      LDX    #TERMES
01410 E241 BD E07E      JSR    PDATA1
01420 E244 86 00      LDA A  #$0
01430 E246 20 4D      BRA    FIN

01450 E248 81 54      CT      CMP A  #'T
01460 E24A 26 C6      BNE    PRTMES      ERR-GO TO START

01480          * TTY SELECTED
01490 E24C 8D 40      WA2    BSR    INCH1
01500 E24E 81 18      CMP A  #$18      TST FOR CTRL X
01510 E250 27 C0      BEQ    PRTMES      GO BACK TO START
01520 E252 81 0D      CMP A  #$D      TST FOR CR
01530 E254 26 F6      BNE    WA2      WAIT FOR CR

```

01540 E256 FF A012	STX	XTEMP		
01550 E259 8D 43	BSR	PRINT	GO PRNT 1ST PART OF MESSAGE	
01560 E25B CE E2F2	LDX	#TTYMES		
01570 E25E BD E07E	JSR	PDATA1		
01580 E261 BD E327	JSR	TTINIT	GO INITIALIZE TTY ACIA	
01590 E264 86 01	LDA A	#\$1	TO MAKE FLAG2 POS	
01600 E266 20 2D	BRA	FIN		
01620 E268 81 50	CP	CMP A	#"P	
01630 E26A 26 A6		BNE	PRTMES	ERR GO TO START
01640 E26C 8D 20		BSR	INCH1	
01650 E26E 81 52		CMP A	#"R	
01660 E270 26 A0		BNE	PRTMES	ERR GO TO START
01680	* PRTR SELECTED			
01690 E272 8D 1A	WA3	BSR	INCH1	
01700 E274 81 18		CMP A	#\$18	TSTS FOR CTRL X
01710 E276 27 9A		BEQ	PRTMES	GO BACK TO START
01720 E278 81 0D		CMP A	#\$D	TST FOR CR
01730 E27A 26 F6		BNE	WA3	WAIT FOR CR
01740 E27C FF A012		STX	XTEMP	
01750 E27F 8D 1D		BSR	PRINT	PRNT 1ST PART OF MESSAGE
01760 E281 CE E2E3		LDX	#PRTRMS	
01770 E284 BD E07E		JSR	PDATA1	
01780 E287 BD E333		JSR	PRINIT	GO INITIALIZE PRTR PIA
01790 E28A 86 FF		LDA A	#\$FF	MAKE FLAG2 NEGATIVE
01800 E28C 20 07		BRA	FIN	
01820 E28E 37	INCH1	PSH B		
01830 E28F BD E1A5		JSR	SAV	
01840 E292 7E E1AF		JMP	IN1	
01860 E295 B7 A015	FIN	STA A	FLAG2	
01870 E298 FE A012		LDX	XTEMP	
01880 E29B 7E E1AC		JMP	\$E1AC	RETURN TO NORMAL INCHAR
01900 E29E CE E304	PRINT	LDX	#PRINT1	
01910 E2A1 BD E07E		JSR	PDATA1	
01920 E2A4 39		RTS		
01940 E2A5 0D	MESS1	FCB	\$D,\$A	
01950 E2A7 53		FCC	/SELECT DESIRED OUTPUT DEVICE - /	
01960 E2C6 54		FCC	/TER, TTY, PRTR/	
01970 E2D4 0D		FCB	\$D,\$A,\$3F,4	
01980 E2D8 54	TERMES	FCC	/TERMINAL/	
01990 E2E0 0D		FCB	\$D,\$A,4	
02000 E2E3 4C	PRTRMS	FCC	/LINE PRINTER/	
02010 E2EF 0D		FCB	\$D,\$A,4	
02020 E2F2 54	TTYMES	FCC	/TELETYPE-PORT 0/	
02030 E301 0D		FCB	\$D,\$A,4	
02040 E304 0D	PRINT1	FCB	\$D,\$A	
02050 E306 41		FCC	/ALL OUTPUT IS NOW ROUTED TO THE /	

02060 E326 04

FCB 4

02080 * THE FOLLOWING SUBROUTINE INITIALIZES THE
 02090 * ACIA WHICH IS USED FOR THE TELETYPE (OR THE
 02100 * H-P 9820 CALCULATOR SYSTEM) AND IS ADDRESSED
 02110 * AT PORT 0 (\$8000)
 02120 E327 CE 8000 TTINIT LDX #\$8000
 02130 E32A 86 13 LDA A #\$13
 02140 E32C A7 00 STA A 0,X
 02150 E32E 86 11 LDA A #\$11
 02160 E330 A7 00 STA A 0,X
 02170 E332 39 RTS

02190 * THE FOLLOWING SUBROUTINE INITIALIZES THE PIA
 02200 * WHICH IS USED FOR THE HEWLETT-PACKARD LINE
 02210 * PRINTER AND IS ADDRESSED AT PORT 7 (\$801C)
 02220 E333 CE 0321 PRINIT LDX #801C
 02230 E336 6F 01 CLR 1,X
 02240 E338 6F 03 CLR 3,X
 02250 E33A 6F 00 CLR 0,X
 02260 E33C 6F 02 CLR 2,X
 02270 E33E 63 00 COM 0,X
 02280 E340 86 3E LDA A #\$3E
 02290 E342 A7 01 STA A 1,X
 02300 E344 86 2E LDA A #\$2E
 02310 E346 A7 03 STA A 3,X
 02320 E348 39 RTS

02340 E349 OCHROU EQU *
 02350 E349 7D A015 TST FLAG2
 02360 E34C 26 07 BNE NZERO

02380 * OUTPUT CHAR TO TERMINAL
 02390 E34E 37 PSH B
 02400 E34F BD E1A5 JSR SAV
 02410 E352 7E E1D4 JMP IOUT

02430 E355 2A 1E NZERO EPL OUTTY

02450 * THE FOLLOWING SUBROUTINE IS THE OUTPUT CHARACTER
 02460 * ROUTINE FOR THE H-P LINE PRINTER AND MAY BE USED
 02470 * THE SAME AS OUTTEE WOULD BE FOR OUTCHAR IN MIKEBUG
 02480 * (ENTRY AT THIS POINT RESULTS IN OUTPUT
 02490 * TO LINE PRINTER REGARDLESS OF STATUS OF FLAG2)

02510 * OUTPUT CHAR TO LINE PTRR
 02520 E357 37 PSH B SAVE ACCB
 02530 E358 FF A012 STX XTEMP SAVE XREG
 02540 E35B 43 COM A PRINTER USES INVERTED LOGIC
 02550 E35C CE 801C LDX #\$801C
 02560 E35F A7 00 STA A 0,X

02570	E361	C6 36	LDA B	#\$36	STROBE DATA READY LINE	
02580	E363	E7 01	STA B	1,X		
02590	E365	C6 3E	LDA B	#\$3E		
02600	E367	E7 01	STA B	1,X		
02610	E369	6D 01	TST	1,X	WAIT FOR HANDSHAKE	
02620	F36B	2A FB	BPL	*-3		
02630	E36D	E6 00	LDA B	0,X		
02640	E36F	43	COM A		RESTORE ACCA TO PREV VALUE	
02650	E370	FE A012	LDX	XTEMP	RESTORE XREG	
02660	E373	33	PUL B		RESTORE B ACCUMULATOR	
02670	E374	39	RTS			
02690	* THE FOLLOWING SUBROUTINE IS THE OUTPUT CHARACTER					
02700	* ROUTINE FOR THE SERIAL PORT WHICH MAY BE CONNECTED					
02710	* TO THE TELETYPE OR TO THE H-P 9820 CALCULATOR SYS					
02720	* THIS SUBROUTINE MAY BE USED THE SAME AS OUTTEE					
02730	* IS USED IN MIKBUG.					
02740	* (ENTRY AT THIS POINT RESULTS IN OUTPUT TO					
02750	* THE SERIAL PORT #0 REGARDLESS OF STATUS OF FLAG2)					
02770	* OUTPUT CHAR TO PORT C ACIA					
02780	E375	37	CUTTY	PSH B		
02790	E376	FF A012	STX	XTEMP		
02800	E379	CE 8000	LDX	#\$8000		
02810	E37C	E6 00	WA4	LDA B	0,X	
02820	E37E	57		ASR B		
02830	E37F	57		ASR B		
02840	E380	24 FA		BCC	WA4	
02850	E382	A7 01		STA A	1,X	
02860	E384	FE A012		LDX	XTEMP	
02870	E387	33		PUL B		
02880	E388	39		RTS		
02900	E389	PWRUP	EQU	*		
02910	E389	CE 8004	LDX	#\$8004		
02920	E38C	7F A014	CLR	FLAG1		
02930	E38F	7F A015	CLR	FLAG2		
02940	E392	7F A016	CLR	FLAG3		
02950	E395	7E E0D9	JMP	\$E0D9		
02980	* ENTER AT BEGINNING OF A MIKBUG "LOAD"					
02990	* THIS ASSURES THE USER THAT AN INADVERTANT					
03000	* CONTROL P ON A TAPE BEING LOADED WILL NOT					
03010	* TERMINATE THE LOADING PROCESS.					
03030	E398	LOAD	EQU	*		
03040	E398	B7 8007	STA A	\$8007		
03050	E39B	B7 A014	STA A	FLAG1	DISABLE ROUTE COMMAND	
03060	E39E	B7 A016	STA A	FLAG3	- INDICATE LOAD IN PROGRESS	
03070	E3A1	39	RTS			

```

03090          * ENTER AT END OF A LOAD TAPE SEQUENCE
03100  E3A2  LODEND EQU  *
03110  E3A2  7D A016    TST    FLAG3  ENABLE ROUTE COMMAND ONLY
03120  E3A5  27 06      BEQ    *+8   IF A LOAD WAS IN PROGRESS
03130  E3A7  7F A016    CLR    FLAG3  INDICATE LOAD NOT IN PROGRESS
03140  E3AA  7F A014    CLR    FLAG1  ENABLE ROUTE COMMAND
03150  E3AB  7E E0E3    JMP    $E0E3  RET TO MIKBUG CONTROL

03170          ****
03180          ****
03190          *
03200          * THE FOLLOWING ROUTINE IS USED ONLY
03210          * IF THE TEST PROGRAM IS LOCATED IN
03220          * EPROM WHICH IS ORG'D AT 'TSTPGM'
03230          *
03240  D000  TSTPGM EQU  $D000
03250          *
03260  E3B0  COMAND EQU  *
03270  E3B0  C1 54      CMP    B  #'T   'T' COMMAND
03280  E3B2  26 03      BNE    *+5
03290  E3B4  7E D000    JMP    TSTPGM  GO TO TEST MONITOR
03300  E3B7  C1 47      CMP    B  #'G
03310  E3B9  26 03      BNE    *+5
03320  E3BB  7E E10F    JMP    $E10F  GO TO USER'S PROGRAM
03330  E3BE  7E E0E3    JMP    $E0F3  RESTART MIKBUG LOOP
03340          *
03350          ****
03360          ****

03380          * THE FOLLOWING IS INCLUDED TO SUPPLY THE
03390          * NECESSARY IRQ, NMI, SWI, AND RESTART VECTORS
03400          * TO THE CPU WHEN THE ADDRESSES $FFF8-$FFFF
03410          * ARE ACCESSED (THE EPROM IN WHICH THIS OVERLAY
03420          * AND MIKBUG ARE LOCATED RESPONDS TO $EXXX-$FXXX)
03430          *
03440  EFFF8          ORG    $EFF8
03450  EFFF8  E000      IRQ    FDB   $E000
03460  EFFA  E005      NMI    FDB   $E005
03470  EFFC  E113      SWI    FDB   $E113
03480  EFFE  E0D0      RESET  FDB   $E0D0

```

03500		END
FLAG1	A014	
FLAG2	A015	
FLAG3	A016	
XTEMP	A012	
PDATA1	E07E	
SAV	E1A5	
IN1	E1AF	
IOUT	E1D4	
ICHROU	E200	
PRTIMES	E212	
WA1	E22F	
CT	E248	
WA2	E24C	
CP	E268	
WA3	E272	
INCH1	E28E	
FIN	E295	
PRINT	E29E	
MESS1	E2A5	
TERMS	E2D8	
PRTIMS	E2E3	
TTYMES	E2F2	
PRINT1	E304	
TTINIT	E327	
PRINIT	E333	
OCHROU	F349	
NZERO	E355	
OUTTY	E375	
WA4	E37C	
PWRUP	E389	
LOAD	E398	
LODEND	E3A2	
TSTPC1	D000	
COMAND	E3B0	
IRQ	EFF8	
NMI	EFFA	
SWI	EFFC	
RESET	EFFE	

TOTAL ERRORS 00000

Appendix B

Test System Software Listing

This Appendix contains the M6800 assembly language source listing for the Test Monitor and all test routines for the NCR 2051 testing system. A discussion of these routines is contained in Chapter V.

00010

NAM NCR.2051

```

00030 ****
00040 ****
00050 ****
00060 ***
00070 *** NCR 2051 WAROM EXORCISER ***
00080 ***
00090 *** WRITTEN BY JOEL W. ROBERTSON ***
00100 ***
00110 *** THIS VERSION DATED ***
00120 ***
00130 *** 17 FEBRUARY 1978 ***
00140 ***
00150 ****
00160 ****
00170 ****

```

00190

OPT NOGEN,O

```

00210 * DEFINE MIKBUG JUMPS
00220 *
00230 E055 INBYTE EQU $E055
00240 E075 OUTCH EQU $E075
00250 E07E PDATA1 EQU $E07E
00260 E0BF OUT2H EQU $E0BF
00270 E0C8 OUT4HS EQU $E0C8
00280 E0CA OUT2HS EQU $E0CA
00290 E1AC INCHAR EQU $E1AC
00300 E0D0 MIKBUG EQU $E0D0

00320 * DEFINE DATA TRANSFER PIA ADDRESSES
00330 *
00340 8010 A1PIAD EQU $8010 ADDR PORT (OUT ONLY)
00350 8014 A2PIAD EQU $8014 CONTRL PORT (OUT ONLY)
00360 8012 B1PIAD EQU $8012 DATA PORT #1 (I/O)
00370 8016 B2PIAD EQU $8016 DATA PORT #2 #1 (I/O)
00380 8016 B2PIAD EQU $8016 DATA PORT #2 (I/O)

00400 * DEFINE CONTROL REGISTER PIA ADDRESSES
00410 *
00420 8011 A1PIAC EQU $8011
00430 8015 A2PIAC EQU $8015
00440 8013 B1PIAC EQU $8013
00450 8017 B2PIAC EQU $8017

```

```

00470      * DEFINE TEMPORARY STORAGE AREAS
00480      *
00490      A050  RDATA1 EQU    $A050  READ DATA MSB
00500      A051  RDATA2 EQU    $A051  READ DATA LSB
00510      A052  WDATA1 EQU    $A052  WRITE DATA MSB
00520      A053  WDATA2 EQU    $A053  WRITE DATA LSB

00540      * STATUS INDICATES WHETHER BPIAD'S ARE PROGRAMMED
00550      * AS INPUTS OR OUTPUTS (0=INPUT)
00560      *
00570      A054  STATUS EQU    $A054
00580      A055  ADDR   EQU    $A055
00590      A056  XTIME  EQU    $A056
00600      A066  EPWID  EQU    $A066
00610      A068  WPWID  EQU    $A068

00630      * WMODE CONTROLS THE WRITE SUBROUTINE
00640      *
00650      * IF WMODE = 0 THEN ERASE BEFORE WRITE
00660      * IF WMODE = + THEN WRITE - NO ERASE
00670      * IF WMODE = - THEN ERASE ONLY
00680      A06D  WMODE  EQU    $A06E

00700 0100      ORG    $0100
00710 0100 7E 01EE  JMP    MON2E  CLEARS WMODE FLAG
00720 0103 7E 01F1  JMP    MON2L2  NO EFFECT ON WMODE

00740      * INITIALIZE ALL PIA'S
00750 0106 7F 8011 INIT   CLR    A1PIAC  ENABLE DDR
00760 0109 7F 8015 CLR    A2PIAC  "
00770 010C 7F 8013 CLR    B1PIAC  "
00780 010F 7F 8017 CLR    B2PIAC  "
00790 0112 7F 8012 CLR    B1PIAD  PROG DATA PORT AS INPUT
00800 0115 7F 8016 CLR    B2PIAD  "      "      "
00820 0118 86 FF  LDA A  #$FF
00830 011A B7 8010 STA A  A1PIAD  PROG ADDR PORT AS OUTPUT
00840 011D B7 8014 STA A  A2PIAD  "      CONTR  "      "
00850 0120 86 04  LDA A  #04
00860 0122 B7 8011 STA A  A1PIAC  ENABLE DATA REGISTERS
00870 0125 B7 8015 STA A  A2PIAC  "      "      "
00880 0128 B7 8013 STA A  B1PIAC  "
00890 012B B7 8017 STA A  B2PIAC  "
00900 012E 7F 8010 CLR    A1PIAD
00910 0131 7F 8014 CLR    A2PIAD
00920 0134 7F A054 CLR    STATUS  -INDICATE DATA PORTS = INPUTS
00930 0137 39  RTS

```

00950 * SUBROUTINE WRITE DATA IN ONE LOCATION
 00960 * ON ENTRY, ASSUME ADDR IN \$A055

00980	0138	7F	8013	WINIT	C,R	B1PIAC	ENABLE DDR
00990	013B	7F	8017		CLR	B2PIAC	ENABLE DDR
01000	013E	C6	FF		LDA B	#\$FF	
01010	0140	F7	8012		STA B	B1PIAD	PROG AS OUTPUTS
01020	0143	F7	8016		STA B	B2PIAD	PROG AS OUTPUTS
01030	0146	F7	8013		STA B	B1PIAC	ENABLE DATA REG
01040	0149	F7	8017		STA B	B2PIAC	ENABLE DATA REG
01050	014C	F7	A054		STA B	STATUS	INDICATE PIA PROG AS OUTPUT
01060	014F	39			RTS		

01080 * THIS SUBROUTINE WRITES DATA (WHICH IS
 01090 * STORED IN \$A052-MSB & \$A053-LSB) INTO
 01100 * THE ADDRESS WHICH IS STORED IN \$A055
 01110 * ERASE TIME DETERMINED BY CONTENTS OF \$A066/7
 01120 * WRITE TIME DETERMINED BY CONTENTS OF \$A068/9

01140	0150	7D	A054	WRITE	TST	STATUS	
01150	0153	26	02		BME	WRITE1	
01160	0155	8D	E1		BSR	WINIT	PREPARE FOR WRITE
01170	0157	B6	A055	WRITE1	LDA A	ADDR	
01180	015A	B7	8010		STA A	A1PIAD	STORE ADDR IN OUTPUT BUFF
01190	015D	B6	A052		LDA A	WDATA1	
01200	0160	F6	A053		LDA B	WDATA2	
01210	0163	F7	8012		STA B	B1PIAD	STORE MSB IN DATA REG
01220	0166	B7	8016		STA A	B2PIAD	STORE LSB IN DATA REG
01230	0169	7D	A06E		TST	WMODE	
01240	016C	27	02		BEQ	BOTH	
01250	016E	2A	0C		BPL	WONLY	WRITE ONLY
01260	0170	86	0E	BOTH	LDA A	#\$0E	
01270	0172	B7	8014		STA A	A2PIAD	ERASE LOCATION
01280				* ERASE TAKES		50-200 MS.	
01290	0175	8D	13		BSR	DELAYE	
01300	0177	7D	A06E		TST	WMODE	
01310	017A	2B	07		BRI	EONLY	ERASE ONLY
01320	017C	86	4A	WONLY	LDA A	#\$4A	1 = WDE,CS ; 0 = C1,C2
01330	017E	B7	8014		STA A	A2PIAD	WRITE DATA IN ADDRESSED LOC
01340	0181	8D	0F		BSR	DELAY	
01350	0183	7F	8014	EONLY	CLR	A2PIAD	0=WDE,CS,C1,C2
01360	0186	7F	8010		CLR	A1PIAD	CLEAR ADDR REG
01370	0189	39			RTS		

```

01390 018A FF A056 DELAYE STX XTEMP
01400 018D FE A066 LDX EPWID
01410 0190 20 06 BRA DEL1
01420 0192 FF A056 DELAY STX XTEMP
01430 0195 F1 A068 LDX WPWID
01440 0198 09 DEL1 DEX
01450 0199 8C 0000 CPX #$0
01460 019C 26 FA BNE DEL1
01470 019E F1 A056 LDX XTEMP
01480 01A1 39 RTS

* SUBROUTINE READ DATA IN 1 LOC
* ASSUMES ADDRESS IN $A055 ON INPUT TO SUBROUTINE
* PREPARE FOR READ

```

```

01530 01A2 7F 8013 RINIT CLR B1PIAC PROG DATA LINES AS INPUTS
01540 01A5 7F 8012 CLR B1PIAD
01550 01A8 7F 8017 CLR B2PIAC
01560 01AB 7F 8016 CLR B2PIAD
01570 01AE 86 04 LDA A #04
01580 01B0 B7 8013 STA A B1PIAC
01590 01B3 B7 8017 STA A B2PIAC
01600 01B6 7F A054 CLR STATUS INDICATE PROGRAMMING = INPUT
01610 01B9 39 RTS

```

```

01630 * THIS SUBROUTINE READS DATA FROM THE
01640 * ADDRESS WHICH IS STORED IN LOCATION
01650 * $A055 AND PLACES THE DATA IN LOCATIONS
01660 * $A050-(MSB) & $A051-(LSB)

```

```

01680 01BA 7D A054 READ TST STATUS
01690 01BD 27 02 BEQ READ1
01700 01BF SD E1 BSR RINIT
01710 01C1 B6 A055 READ1 LDA A ADDR
01720 01C4 B7 8010 STA A A1PIAD STORE ADDR IN OUTPUT BUFFER
01730 01C7 86 02 LDA A #02
01740 01C9 B7 8014 STA A A2PIAD ENABLE CHIP SELECT (CS)
01750 01CC 86 03 LDA A #03
01760 01CE B7 8014 STA A A2PIAD START CLOCK PULSE
01770 01D1 C6 02 LDA B #$02

```

```

01790 * GEN 5 CPU CYCLE CLK PULSE
01800 01D3 F7 8014 STA B A2PIAD STOP CLK
01810 01D6 C6 32 LDA B #$32
01820 01D8 F7 8014 STA B A2PIAD STROBE DATA & ENABLE
01830 * READ DATA SIGNAL (RDE)
01840 01DB F6 8012 LDA B B1PIAD READ MSB OF DATA
01850 01DE B6 8016 LDA A B2PIAD READ LSB OF DATA
01860 01E1 B7 A050 STA A RDATA1
01870 01E4 F7 A051 STA B RDATA2 PUT DATA IN TEMP STORAGE
01880 01E7 7F 8014 CLR A2PIAD 0=CS, RDE, DATA STROBE
01890 01EA 7F 8010 CLR A1PIAD 0=ADDRESS BUFFER
01900 01ED 39 RTS

```

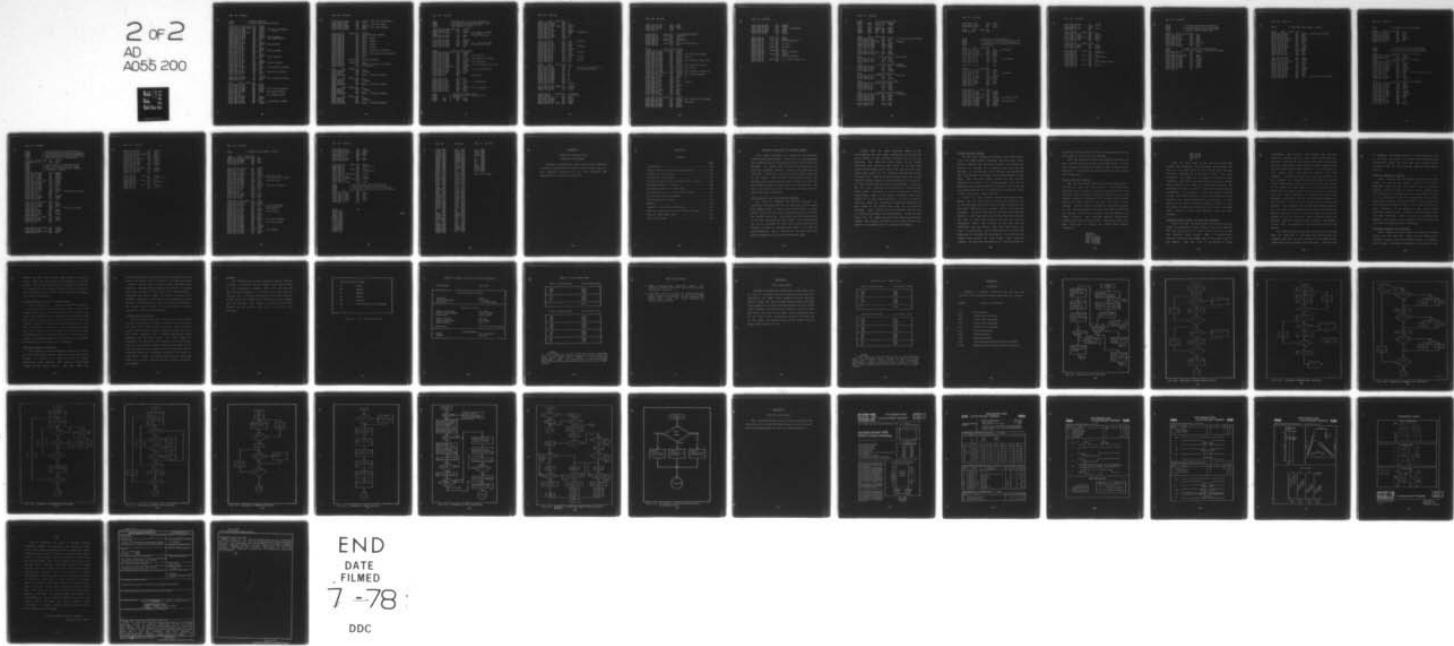
AD-A055 200 AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OHIO SCH--ETC F/G 14/2
DESIGN OF A MICROPROCESSOR BASED SYSTEM FOR TESTING OF THE NCR --ETC(U)
MAR 78 J W ROBERTSON

UNCLASSIFIED

AFIT/GE/EE/78-7

NL

2 of 2
AD
A055 200



END

DATE

FILMED

7-78

DDC

01920	* MONITOR NUMBER TWO		
01930	* ENTER FOR TEST PROGRAM SELECTION		
01950 01EE 7F A06D	MON2E	CLR	WMODE
01960 01F1 CE 1000	MON2E2	LDX	#\$1000
01970 01F4 FF A068		STX	WPVID
01980 01F7 FF A066		STX	EPVID
01990 01FA 8E A040	MON2	LDS	#\$A040
02000 01FD BD 0106		JSR	INIT
02010 0200 CE 0279		LDX	#COMM16
02020 0203 BD E07E		JSR	PDATA1
02030 0206 BD 03D7		JSR	INHEX
02040 0209 81 01		CMP A	#1
02050 020B 26 05		BNE	*+7
02060 020D CE 0349		LDX	#MARCH1
02070 0210 20 33		BRA	EXMON
02080 0212 81 02		CMP A	#2
02090 0214 26 05		BNE	*+7
02100 0216 CE 0354		LDX	#MASES1
02110 0219 20 2A		BRA	EXMON
02120 021B 81 03		CMP A	#3
02130 021D 26 05		BNE	*+7
02140 021F CE 0360		LDX	#GALPA1
02150 0222 20 21		BRA	EXMON
02160 0224 81 04		CMP A	#4
02170 0226 26 05		BNE	*+7
02180 0228 CE 036C		LDX	#WAKPA1
02190 022B 20 18		BRA	EXMON
02200 022D 81 FF		CMP A	#\$FF
02210 022F 26 03		BNE	*+5
02220 0231 7E 04A2		JMP	PLSE WIDTH CTRL SELECTED
02230 0234 81 0D		CMP A	#\$D
02240 0236 26 03		BNE	*+5
02250 0238 BD 0693		JSR	DISPLAY MNOS CONTENTS
02260 023B 20 BD		BRA	MON2
02280 023D CE 0326		LDX	#COMM18
02290 0240 BD E07E		JSR	PRNT 'INVALID SELECTION'
02300 0243 20 B5		BRA	MON2
02320 0245 EXMON		EQU	*
02330 0245 FF A056		STX	XTEMP
02340 0248 BD 0450		JSR	OCRLF
02350 024B CE 0314		LDX	#COMM17
02360 024E BD E07E		JSR	PDATA1
02370 0251 FE A056		LDX	XTEMP
02380 0254 BD E07E		JSR	PDATA1
02390 0257 08		INX	PRNT NAME OF PROGRAM
02400 0258 EE 00		LDX	X
02410 025A FF A056		STX	GET SUBROUTINE ADDR
02420 025D 8C 0714		CPX	XTEMP
02430 0260 27 03		BEQ	#MASEST
02440 0262 BD 0396		JSR	*+5
02450 0265 CE 0558		START	GET TESTWORD & PATTERN
		LDX	#COMM24

02460 0268 BD E07E	JSR	PDATA1	PRNT 'TEST IN PROGRESS'
02470 026B FE A056	LDX	XTEMP	
02480 026E AD 00	JSR	X	GO TO TEST SUBRT
02490 0270 CE 056B	LDX	#COMM25	
02500 0273 BD E07E	JSR	PDATA1	PRNT 'TEST COMPLETE'
02510 0276 7E 01FA	JMP	MON2	
02530 0279 0D	COMM116	FCB	\$0D,\$0A,
02540 027C 53		FCC	/SELECT DESIRED PROGRAM/
02550 0292 0D		FCB	\$D,\$A
02560 0294 30		FCC	/01 MARCH/
02570 02A0 0D		FCB	\$D,\$A
02580 02A2 30		FCC	/02 MASEST/
02590 02AF 0D		FCB	\$D,\$A
02600 02B1 30		FCC	/03 GALPAT/
02610 02BE 0D		FCB	\$D,\$A
02620 02C0 30		FCC	/04 WAKPAT/
02630 02CD 0D		FCB	\$D,\$A
02640 02CF 30		FCC	/0D DISPLAY MNOS CONTENTS/
02650 02EB 0D		FCB	\$D,\$A
02660 02ED 46		FCC	"FF PULSE WIDTHS DISPLAY/CHANGE"
02670 030F 0D		FCB	\$D,\$A
02680 0311 3F		FCC	/? /
02690 0313 04		FCB	4
02700 0314 50	COMM117	FCC	/PROGRAM SELECTED /
02710 0325 04		FCB	4
02720 0326 0D	COMM118	FCB	\$D,\$A
02730 0328 49		FCC	/INVALID SELECTION - TRY AGAIN/
02740 0345 0D		FCB	\$D,\$A,\$3F,4
02760 0349	MARCH1	EQU	*
02770 0349 0D		FCB	\$D,\$A
02780 034B 20		FCC	/ MARCH/
02790 0351 04		FCB	4
02800 0352 06C6		FDB	MARCH SUBROUTINE ADDRESS
02810	*		
02820 0354	MASEST1	EQU	*
02830 0354 0D		FCB	\$D,\$A
02840 0356 20		FCC	/ MASEST/
02850 035D 04		FCB	4
02860 035E 0714		FDB	MASEST SUBROUTINE ADDRESS
02870	*		
02880 0360	GALPA1	EQU	*
02890 0360 0D		FCB	\$D,\$A
02900 0362 20		FCC	/ GALPAT/
02910 0369 04		FCB	4
02920 036A 0825		FDB	GALPAT SUBROUTINE ADDRESS
02930	*		
02940 036C	WAKPA1	EQU	*
02950 036C 0D		FCB	\$D,\$A
02960 036E 20		FCC	/ WAKPAT/
02970 0375 04		FCB	4
02980 0376 0583		FDB	WAKPAT SUBROUTINE ADDRESS

```

03000          * THIS SUBROUTINE PRINTS THE CONTENTS OF
03010          * THE BACKGROUND TEST PATTERN AND TEST WORD
03020          * AND ALLOWS THE OPERATOR TO CHANGE
03030          * THE CONTENTS IF DESIRED

03050 0378 CE 045B RSTART LDX    #COMM11
03060 037B BD 0450      JSR    OCRLF
03070 037E BD E07E      JSR    PDATA1  PRNT 'INPUT BACKGROUND
03080          *          TTEST PATTERN'
03090 0381 BD 03D7      JSR    INHEX   GO INPUT 4 HEX CHAR
03100 0384 FF A064      STX    BEE

03120 0387 CE 047A      LDX    #COMM12
03130 038A BD 0450      JSR    OCRLF
03140 038D BD E07E      JSR    PDATA1  PRNT "INPUT TEST WORD"
03150 0390 BD 03D7      JSR    INHEX   GO INPUT 4 HEX CHAR
03160 0393 FF A062      STX    TEE

03180          * ENTER SUBROUTINE HERE
03190 0396 CE 0471 START  LDX    #COMM13
03200 0399 BD 0450      JSR    OCRLF
03210 039C BD E07E      JSR    PDATA1  PRNT "PATTERN "
03220 039F CE A064      LDX    #BEE
03230 03A2 BD E0C8      JSR    OUT4HS  PRNT CONTENTS OF PATTERN
03240 03A5 CE 0480      LDX    #COMM14
03250 03A8 BD E07E      JSR    PDATA1  PRNT "TEST WORD "
03260 03AB CE A062      LDX    #TIE
03270 03AE BD E0C8      JSR    OUT4HS  GO PRNT CONTENTS OF TEST WORD
03280 03B1 CE 048B DECIS LDX    #COMM15
03290 03B4 BD 0450      JSR    OCRLF
03300 03B7 BD E07E      JSR    PDATA1  PRNT "GO OR RE-ENTER ? "
03310 03BA BD E1AC      JSR    INCHAR
03320 03BD 81 47      CMP A  #'G
03330 03BF 26 01      BNE    *+3
03340 03C1 39          RTS    GO RUN TEST

03360 03C2 81 52      CMP A  #'R
03370 03C4 26 03      BNE    *+5
03380 03C6 7E 0378      JMP    RSTART  GO RE-ENTER DATA
03390 03C9 81 58      CMP A  #'X
03400 03CB 26 03      BNE    *+5
03410 03CD 7E E0D0      JMP    MIKBUG
03420 03D0 81 03      CMP A  #\$03  TST FOR CONTROL C
03430 03D2 26 DD      BNE    DECIS
03440 03D4 7E 01FA      JMP    MON2

03460          * INPUT HEX CHARACTER STRING
03470          * TERMINATED BY CR, X, CONTROL X, OR
03480          * CONTROL C
03490      A061  H      EQU    \$A061
03500      A060  K      EQU    \$A060

```

03520	03D7	INHEX	EQU	*		
03530	03D7	7F A061	CLR	H		
03540	03DA	7F A060	CLR	K		
03550	03DD	BD E1AC	INHEX1	JSR	INCHAR	
03560	03E0	81 58	CMP A	#'X		
03570	03E2	26 03	BNE	*+5		
03580	03E4	7E E0D0	JMP	MIKBUG		
03590	03E7	81 0D	CMP A	#\$0D	CARRIAGE RET	
03600	03E9	26 07	BNE	*+9		
03610	03EB	FE A060	LDX	K		
03620	03EE	B6 A061	LDA A	H		
03630	03F1	39	RTS			
03650	03F2	81 03	CMP A	#\$03	CONTROL C	
03660	03F4	26 03	BNE	*+5		
03670	03F6	7E 01FA	JMP	MON2		
03680	03F9	81 18	CMP A	#\$18	CONTROL X	
03690	03FB	26 03	BNE	*+5		
03700	03FD	7E 0432	JMP	DELETE		
03710	0400	80 30	SUB A	#\$30		
03720	0402	2B 41	BMI	C1	NOT HEX	
03730	0404	81 09	CMP A	#\$09		
03740	0406	2F 08	BLE	IN1HG		
03750	0408	2B 3B	BMI	C1	NOT HEX	
03760	040A	81 16	CMP A	#\$16		
03770	040C	2E 37	BGT	C1	NOT HEX	
03780	040E	80 07	SUB A	#\$7		
03800	0410	IN1HG	EQU	*		
03810	0410	84 0F	AND A	#\$0F		
03820	0412	78 A061	ASL	H	SHIFT 2 BYTE REGISTER (H-K)	
03830	0415	79 A060	ROL	K	ONE CHARACTER TO LEFT	
03840	0418	78 A061	ASL	H		
03850	041B	79 A060	ROL	K		
03860	041E	78 A061	ASL	H		
03870	0421	79 A060	ROL	K		
03880	0424	78 A061	ASL	H		
03890	0427	79 A060	ROL	K		
03900	042A	BB A061	ADD A	H		
03910	042D	B7 A061	STA A	H		
03920	0430	20 AB	BRA	INHEX1		
03940	0432	CE 043A	DELETE	LDX	#COMM10	
03950	0435	BD E07E	JSR	PDATA1		
03960	0438	20 9D	BRA	INHEX		
03980	043A	20	COMM10	FCC	/ DELETED/	
03990	0442	0D		FCB	\$0D,\$0A,\$04	
04000	0445	86 3F	C1	LDA A	#'?	
04010	0447	BD E075	JSR	OUTCH		
04020	044A	BD 0450	JSR	OCRLF		
04030	044D	7E 0432	JMP	DELETE		
04040	0450	86 0D	OCRLF	LDA A	#\$0D	

04050 0452 BD E075	JSR	OUTCH
04060 0455 86 0A	LDA A	#\$0A
04070 0457 BD E075	JSR	OUTCH
04080 045A 39	RTS	
04100 045B 49	COMM11 FCC	/INPUT BACKGROUND TEST /
04110 0471 50	COMM13 FCC	/PATTERN /
04120 0479 04	FCB	\$04
04130 047A 49	COMM12 FCC	/INPUT /
04140 0480 54	COMM14 FCC	/TEST WORD /
04150 048A 04	FCB	\$04
04160 048B 47	COMM15 FCC	/GO OR RE-ENTER DATA ? /
04170 04A1 04	FCB	\$04
04190	*	SUBPROGRAM PCONTL
04200 04A2 CE 051F	PCONTL LDX	#COMM22
04210 04A5 BD 0450	JSR	OCRLF
04220 04A8 BD E07E	JSR	PDATA1 PRNT "PULSE WIDTH WORD"
04230 04AB CE 0531	LDX	#COMM23
04240 04AE BD E07E	JSR	PDATA1 "WRITE"
04250 04B1 CE A068	LDX	#WPVID
04260 04B4 BD E0C8	JSR	OUT4HS PRNT CONTENTS OF WRITE WORD
04270 04B7 CE 051F	LDX	#COMM22
04280 04BA BD 0450	JSR	OCRLF
04290 04BD BD E07E	JSR	PDATA1 PRNT "PULSE WIDTH WORD"
04300 04C0 CE 053A	LDX	#COMM19
04310 04C3 BD E07E	JSR	PDATA1 PRNT "ERASE"
04320 04C6 CE A066	LDX	#EPVID
04330 04C9 BD E0C8	JSR	OUT4HS PRNT CONTENTS OF ERASE WORD
04340 04CC CE 0543	DECS	LDX #COMM20
04350 04CF BD E07E	JSR	PDATA1 PRNT "OK OR RE-ENTER?"
04360 04D2 BD E1AC	JSR	INCHAR
04370 04D5 81 4F	CMP A	#"O OK
04380 04D7 26 03	BNE	*+5
04390 04D9 7E 01FA	JMP	MON2
04400 04DC 81 52	CMP A	#"R RE-ENTER
04410 04DE 26 03	BNE	*+5
04420 04E0 7E 04F1	JMP	CHANGE
04430 04E3 81 58	CMP A	#"X
04440 04E5 26 03	BNE	*+5
04450 04E7 7E E0D0	JMP	HIKBUG
04460 04EA 81 03	CMP A	#\$3
04470 04EC 26 DE	BNE	DECS
04480 04EE 7E 01FA	JMP	MON2
04500 04F1 CE 0517	CHANGE LDX	#COMM21
04510 04F4 BD E07E	JSR	PDATA1 PRNT "INPUT PULSE WIDTH WORD"
04520 04F7 CE 0531	LDX	#COMM23
04530 04FA BD E07E	JSR	PDATA1 PRNT "WRITE"
04540 04FD BD 03D7	JSR	INHEX
04550 0500 FF A068	STX	WPVID
04560 0503 CE 0517	LDX	#COMM21

04570 0506 BD E07E	JSR	PDATA1
04580 0509 CE 053A	LDX	#COMM19
04590 050C BD E07E	JSR	PDATA1 PRNT"ERASE"
04600 050F BD 03D7	JSR	INHEX
04610 0512 FF A066	STX	EPWID
04620 0515 20 8B	BRA	PCONTL
04640 0517 0D	COMM21	FCB \$D,\$A
04650 0519 49	FCC	/INPUT /
04660 051F 50	COMM22	FCC /PULSE WIDTH WORD /
04670 0530 04	FCB	4
04680 0531 2D	COMM23	FCC /- WRITE /
04690 0539 04	FCC	4
04700 053A 2D	COMM19	FCC /- ERASE /
04710 0542 04	FCB	4
04720 0543 0D	COMM120	FCB \$D,\$A
04730 0545 4F	FCC	/OK OR RE-ENTER ?/
04740 0555 0D	FCB	\$D,\$A,4
04750 0558 0D	COMM124	FCB \$D,\$A
04760 055A 54	FCC	/TEST IN PROGRESS/
04770 056A 04	FCB	4
04780 056B 20	COMM125	FCC / --- TEST COMPLETE ---/
04790 0582 04	FCB	4

```

04810      * DEFINE SCRATCH AREAS
04820  A06A  PTR    EQU    $A06A
04830  A06A  CURADD EQU    $A06A
04840  A06B  COMADD EQU    $A06B
04850  A06C  CDATA1 EQU    $A06C
04860  A062  TEE    EQU    $A062
04870  A064  BEE    EQU    $A064

04890  0583  WAKPAT EQU    *
04900  0583  BD 05E3  JSR    BCKGND  GO WRITE BACKGROUND PATTERN
04910  0586  7F A06A  CLR    PTR
04920  0589  FE A062  LOOP1  LDX    TEE    TESTWORD
04930  058C  FF A052  STX    WDATA1
04940  058F  B6 A06A  LDA A  PTR
04950  0592  B7 A055  STA A  ADDR
04960  0595  BD 0150  JSR    WRITE

04980  0598  7F A055  CLR    ADDR
04990  059B  BD 01BA  LOOP2  JSR    READ
05000  059E  FE A050  LDX    RDATA1

05020      * TST FOR LOCATION WITH TEST WORD
05030  05A1  B6 A055  LDA A  ADDR
05040  05A4  B1 A06A  CMP A  PTR
05050  05A7  27 0A    BEQ    LOOP2A

05070      * TEST FOR BACKGROUND PATTERN
05080  05A9  BC A064  CPX    BEL    BACKGROUND
05090  05AC  27 03    BEQ    *+5
05100  05AE  BD 0619  JSR    ERROR1
05110  05B1  20 08    BRA    LOOP2B

05130      * TEST FOR TEST WORD
05140  05B3  BC A062  LOOP2A CPX    TEE    TEST WORD
05150  05B6  27 03    BEQ    *+5
05160  05B8  BD 0640  JSR    ERROR2

05180      * TST FOR TOP OF MEMORY
05190  05BB  B6 A055  LOOP2B LDA A  ADDR
05200  05BE  81 1F    CMP A  #$1F
05210  05C0  27 05    BEQ    LOOP3
05220  05C2  7C A055  INC    ADDR
05230  05C5  20 D4    BRA    LOOP2

05250      * RESTORE BACKGROUND PATTERN
05260  05C7  FE A064  LOOP3  LDX    BEE    BACKGROUND
05270  05CA  FF A052  STX    WDATA1
05280  05CD  B6 A06A  LDA A  PTR
05290  05D0  B7 A055  STA A  ADDR
05300  05D3  BD 0150  JSR    WRITE

05320  05D6  B6 A06A  LDA A  PTR
05330  05D9  81 1F    CMP A  #$1F

```

05340 05DB 27 05		BEQ	ENDIT
05350 05DD 7C A06A		INC	PTR
05360 05E0 20 A7		BRA	LOOP1
05380 05E2	ENDIT	EQU	*
05390 05E2 39		RTS	

05410 * SUBROUTINE WRITE BACKGROUND
 05420 * THIS WRITES THE BACKGROUND PATTERN IN THE
 05430 * ENTIRE MEMORY THEN READS THE MEMORY
 05440 * AND VERIFIES THAT THE BACKGROUND PATTERN
 05450 * WAS WRITTEN

05470 05E3	BCKGND	EQU	*
05480 05E3 FE A064		LDX	BEE BACKGROUND
05490 05E6 FF A052		STX	WDATA1
05500 05E9 7F A055		CLR	ADDR
05510 05EC BD 0150	BCK1	JSR	WRITE
05520 05EF B6 A055		LDA A	ADDR
05530 05F2 81 1F		CMP A	#\$1F
05540 05F4 27 05		BEQ	*+7 TOP OF MEMORY
05550 05F6 7C A055		INC	ADDR
05560 05F9 20 F1		BRA	BCK1

05580 05FB 7F A055		CLR	ADDR
--------------------	--	-----	------

05600	*	VERIFY	
05610 05FE BD 01BA	BCK2	JSR	READ
05620 0601 FE A050		LDX	RDATA1
05630 0604 BC A064		CPX	BEE BACKGROUND
05640 0607 27 03		BEQ	*+5
05650 0609 BD 064F		JSR	ERROR3
05660 060C B6 A055		LDA A	ADDR
05670 060F 81 1F		CMP A	#\$1F
05680 0611 27 05		BEQ	*+7
05690 0613 7C A055		INC	ADDR
05700 0616 20 E6		BRA	BCK2
05710 0618 39		RTS	

05730 0619 36	ERROR1	PSH A	
05740 061A FF A056		STX	XTEMP
05750 061D CE 0658		LDX	#CERR1
05760 0620 BD E07E	ERR1A	JSR	PDATA1
05770 0623 CE A064		LDX	#EPE
05780 0626 BD E0C8	ERR1B	JSR	OUT4HS PRINT WRITTEN DATA
05790 0629 CE A050		LDX	#RDAT1
05800 062C BD E0C8		JSR	OUT4HS PRINT READ DATA
05810 062F CE 068B		LDX	#CADD
05820 0632 BD E07E		JSR	PDATA1
05830 0635 CE A055		LDX	#ADDR

05840 0638 BD E0CA	JSR	OUT2HS
05850 063B 32	PUL A	
05860 063C FE A056	LDX	XTEMP
05870 063F 39	RTS	
05890 0640 36	ERROR2 PSH A	
05900 0641 FF A056	STX	XTEMP
05910 0644 CE 0666	LDX	#CERR2
05920 0647 BD E07E	ERR2A JSR	PDATA1
05930 064A CE A062	LDX	#TLE
05940 064D 20 D7	BRA	ERR1B
05960 064F 36	ERROR3 PSH A	
05970 0650 FF A056	STX	XTEMP
05980 0653 CE 0675	LDX	#CERR3
05990 0656 20 C8	BRA	ERR1A
06010 0658 0D	CERR1 FCB	\$D,\$A
06020 065A 45	FCC	/ERR-TYPE 1 /
06030 0665 04	FCB	4
06040 0666 0D	CERR2 FCB	\$D,\$A
06050 0668 45	FCC	/ERR- TYPE 2 /
06060 0674 04	FCB	4
06070 0675 0D	CERR3 FCB	\$D,\$A
06080 0677 45	FCC	/ERR- BCKGND VERIFY /
06090 068A 04	FCB	4
06100 068B 41	CADD FCC	/ADDR = /
06110 0692 04	FCB	4

06130 * SUBROUTINE DISPLAY THE CONTENTS OF
 06140 * THE MNOS MEMORY ON THE CRT AND WAIT
 06150 * FOR ANY CHARACTER TO BE INPUT BEFORE
 06160 * RETURNING TO THE CONTROL LOOP.

06180	0693	DISPLY	EQU	*		
06190	0693	7F	A055	CLR	ADDR	
06200	0696	8D	12	BSR	DIS1	
06210	0698	8D	13	BSR	DIS2	
06220	069A	B6	A055	LDA	A ADDR	
06230	069D	81	20	CMP	A #\$20	
06240	069F	26	F5	BNE	DISPLY+3	
06250	06A1	86	3F	LDA	A #'?	
06260	06A3	BD	E075	JSR	OUTCH	
06280	* DISPLAY CONTENTS OF MNOS MEMORY ON					
06290	* CRT UNTIL ANY CHAR IS INPUT FROM KEYBOARD					
06300	06A6	BD	E1AC	JSR	INCHAR	
06310	06A9	39		RTS		
06320	06AA	BD	0450	DIS1	JSR	OCRLF
06330	06AD	BD	01BA	DIS2	JSR	READ
06340	06B0	CE	A050		LDX	#RDATA1
06350	06B3	BD	E0C8		JSR	OUT4HS
06360	06B6	CE	068B		LDX	#CADD
06370	06B9	BD	E07E		JSR	PDATA1
06380	06BC	CE	A055		LDX	#ADDR
06390	06BF	BD	E0CA		JSR	OUT2HS
06400	06C2	7C	A055		INC	ADDR
06410	06C5	39		RTS		

06440

* SUBROUTINE TEST PROGRAM -- MARCH

06460	06C6	MARCH	EQU	*	
06470	06C6	BD 05E3	JSR	BCKGND	WRITE BCKGND PATTERN
06480	06C9	7F A055	CLR	ADDR	
06490	06CC	FE A062	LDX	TEE	
06500	06CF	FF A052	STX	WDATA1	
06510	06D2	BD 01BA LP1	JSR	READ	
06520	06D5	FE A050	LDX	RDATA1	
06530	06D8	BC A064	CPX	BEE	
06540	06DB	27 03	BEQ	*+5	
06550	06DD	BD 0619	JSR	ERROR1	
06560	06E0	BD 0150	JSR	WRITE	
06570	06E3	B6 A055	LDA A	ADDR	
06580	06E6	81 1F	CMP A	#\$1F	
06590	06E8	27 05	BEQ	*+7	
06600	06EA	7C A055	INC	ADDR	
06610	06ED	20 E3	BRA	LP1	
06630	06EF	FE A064	LDX	BEE	
06640	06F2	FF A052	STX	WDATA1	
06650	06F5	BD 01BA LP2	JSR	READ	
06660	06F8	FE A050	LDX	RDATA1	
06670	06FB	BC A062	CPX	TEE	
06680	06FE	27 03	BEQ	*+5	
06690	0700	BD 0640	JSR	ERROR2	
06700	0703	BD 0150	JSR	WRITE	
06710	0706	7D A055	TST	ADDR	
06720	0709	27 05	BEQ	*+7	
06730	070B	7A A055	DEC	ADDR	
06740	070E	20 E5	BRA	LP2	
06750	0710	BD 05FE	JSR	BCK2	VERIFY BCKGND RE-WRITTEN
06760	0713	39	RTS		

06780 * SUBROUTINE TEST PROGRAM MASEST

```

06300 0714 MASEST EQU *
06810 0714 BD 0727 JSR PRSET1
06820 0717 BD 0773 JSR TEST
06830 071A BD 074F JSR VFY
06840 071D BD 072F JSR PRSET2
06850 0720 BD 0773 JSR TEST
06860 0723 BD 074F JSR VFY
06870 0726 39 RTS

```

```

06890 * ON RETURN FROM THIS SUBROUTINE THE
06900 * DATA IN WDATA1 IS THE DATA WHICH WAS
06910 * WRITTEN INTO ALL EVEN MEMORY WORDS

```

```

06930 * START WITH ALL 1'S IN ADDR 00
06940 0727 CE FFFF PRSET1 LDX #$FFFF
06950 072A FF A052 STX WDATA1
06960 072D 20 06 BRA PRSET
06970 * START WITH ALL 0'S IN ADDR 00
06980 072F CE 0000 PRSET2 LDX #0
06990 0732 FF A052 STX WDATA1
07000 0735 7F A055 PRSET CLR ADDR
07010 0738 BD 0150 JSR WRITE
07020 073B 73 A052 COM WDATA1
07030 073E 73 A053 COM WDATA1+1
07040 0741 B6 A055 LDA A ADDR
07050 0744 81 1F CMP A #$1F TEST FOR TOP OF MEMORY
07060 0746 26 02 BNE *+4
07070 0748 20 05 BRA VFY
07080 074A 7C A055 INC ADDR
07090 074D 20 E9 BRA PRSET+3

```

```

07110 * VERIFY ALTERNATING DATA PATTERN
07120 074F 7F A055 VFY CLR ADDR
07130 0752 BD 01BA JSR READ
07140 0755 FE A050 LDX RDATA1
07150 0758 BC A052 CPX WDATA1
07160 075B 27 03 BEQ *+5
07170 075D BD 07CA JSR ERROR6
07180 0760 73 A052 COM WDATA1
07190 0763 73 A053 COM WDATA1+1
07200 0766 B6 A055 LDA A ADDR
07210 0769 81 1F CMP A #$1F
07220 076B 26 01 BNE *+3
07230 076D 39 RTS
07240 076E 7C A055 INC ADDR
07250 0771 20 DF BRA VFY+3

```

07280 * THIS SUBROUTINE TESTS THE MEMORY FOR
 07290 * ALTERNATING DATA PATTERNS (DATA-COMPLEMENT-
 07300 * DATA-COMPLEMENT-ETC.) ASSUMING THAT THE
 07310 * DATA IN WDATA1 UPON ENTRY IS THE DATA WHICH
 07320 * WAS WRITTEN INTO ALL EVEN ADDRESSES.

 07340 0773 7F A06A TEST CLR CURADD
 07350 0776 86 1F LDA A #\$1F
 07360 *
 07370 * COMADD CONTAINS THE COMPLEMENT OF THE
 07380 * ADDRESS IN WHICH THE DATA WAS WRITTEN
 07390 * (THE ADDRESS IN WHICH DATA WAS WRITTEN
 07400 * IS CONTAINED IN CURADD)
 07410 *
 07420 0778 B7 A06B STA A COMADD
 07430 077B FE A052 LDX WDATA1
 07440 077E FF A06C STX CDATA1
 07450 0781 73 A06C COM CDATA1
 07460 0784 73 A06D COM CDATA1+1
 07470 0787 B6 A06A TST2 LDA A CURADD
 07480 078A BD 07A9 JSR TST1
 07490 078D B6 A06B LDA A COMADD
 07500 0790 BD 07A9 JSR TST1
 07510 0793 B6 A06A LDA A CURADD
 07520 0796 BD 07A9 JSR TST1
 07530 0799 B6 A06A LDA A CURADD
 07540 079C 81 1F CMP A #\$1F TEST FOR TOP OF MEMORY
 07550 079E 26 01 BNE *+3
 07560 07A0 39 RTS
 07570 07A1 7C A06A INC CURADD
 07580 07A4 7A A06B DEC COMADD
 07590 07A7 20 DE BRA TST2

 07610 07A9 B7 A055 TST1 STA A ADDR
 07620 07AC BD 01BA JSR READ
 07630 07AF FE A050 LDX RDATA1
 07640 07B2 B6 A055 LDA A ADDR
 07650 07B5 47 ASR A TST FOR ODD OR EVEN
 07660 07B6 24 09 BCC EVEN
 07670 07B8 BC A06C CPX CDATA1
 07680 07BB 27 03 BEQ *+5
 07690 07BD BD 07CF JSR ERROR4
 07700 07C0 39 RTS
 07710 07C1 BC A052 EVEN CPX WDATA1
 07720 07C4 27 03 BEQ *+5
 07730 07C6 BD 07D9 JSR ERRCR5
 07740 07C9 39 RTS

 07760 07CA CE 0817 ERROR6 LDX #CERR6
 07770 07CD 20 0D BRA ERROR5+3
 07780 07CF CE 07FB ERROR4 LDX #CERR4
 07790 07D2 8D 1D BSR E3A

07800	07D4	CE A06C	LDX	#CDATA1
07810	07D7	20 08	BRA	E3B
07820	07D9	CE 0809	ERROR5	LDX #CERR5
07830	07DC	8D 13	BSR	E3A
07840	07DE	CE A052	LDX	#RDATA1
07850	07E1	BD E0C8	E3B JSR	OUT4HS
07860	07E4	CE 068B	LDX	#CADD
07870	07E7	BD E07E	JSR	PDATA1
07880	07EA	CE A055	LDX	#ADDR
07890	07ED	BD E0CA	JSR	OUT2HS
07900	07F0	39	RTS	
07910	07F1	BD E07E	E3A JSR	PDATA1
07920	07F4	CE A050	LDX	#RDATA1
07930	07F7	BD E0C8	JSR	OUT4HS
07940	07FA	39	RTS	
07960	07FB	0D	CERR4 FCB	\$D,\$A
07970	07FD	45	FCC	/ERR-TYPE 4 /
07980	0808	04	FCB	4
07990	0809	0D	CERR5 FCB	\$D,\$A
08000	080B	45	FCC	/ERR-TYPE 5 /
08010	0816	04	FCB	4
08020	0817	0D	CERR6 FCB	\$D,\$A
08030	0819	45	FCC	/ERR-TYPE 6 /
08040	0824	04	FCB	4

08060

* SUBROUTINE TEST PROGRAM -- GALPAT

08080	0825	GALPAT	EQU	*	
08090	0825	BD	0832	GALPAT	JSR AA
08100	0828	BD	08E5		JSR SWAP
08110	082B	BD	0832		JSR AA
08120	082E	BD	08E5		JSR SWAP
08130	0831	39			RTS
08150	0832	BD	05E3	AA	JSR BCKGND
08160	0835	7F	A06A		CLR PTR
08170	0838	7F	A055		CLR ADDR
08180	083B	FE	A062	A1	LDX TEE
08190	083E	FF	A052		STX WDATA1
08200	0841	BD	0150		JSR WRITE (WRITE TEST WORD)
08210	0844	BD	0869		JSR GTEST GO TO READ-VERIFY SEQUENCE
08220	0847	B6	A06A		LDA A PTR
08230	084A	B7	A055		STA A ADDR
08240	084D	FE	A064		LDX BEE
08250	0850	FF	A052		STX WDATA1
08260	0853	BD	0150		JSR WRITE (RESTORE BACKGROUND)
08270	0856	B6	A06A		LDA A PTR
08280	0859	81	1F		CMP A #\$1F
08290	085B	26	01		BNE *+3
08300	085D	39			RTS
08310	085E	7C	A06A		INC PTR
08320	0861	B6	A06A		LDA A PTR
08330	0864	B7	A055		STA A ADDR
08340	0867	20	D2		BRA A1
08360	0869	7F	A055	GTEST	CLR ADDR
08370	086C	B6	A055	A2	LDA A ADDR
08380	086F	B7	A06B		STA A COMADD
08390	0872	B1	A06A		CMP A PTR TST FOR TSTWORD LOC
08400	0875	27	36		BEQ A3 DO NOT READ & CMP
08410	0877	BD	01BA		JSR READ (READ BCKGND)
08420	087A	FE	A050		LDX RDATA1
08430	087D	DC	A064		CPX BEE TST FOR BCKGND
08440	0880	27	03		BEQ *+5
08450	0882	BD	08D7		JSR ERROR7
08460	0885	B6	A06A		LDA A PTR
08470	0888	B7	A055		STA A ADDR
08480	088B	BD	01BA		JSR READ READ TSTWORD ADDRESS
08490	088E	FE	A050		LDX RDATA1
08500	0891	BC	A062		CPX TEE COMP WITH TESTWORD
08510	0894	27	03		BEQ *+5
08520	0896	BD	08DE		JSR ERROR8
08530	0899	B6	A06B		LDA A COMADD
08540	089C	B7	A055		STA A ADDR
08550	089F	BD	01BA		JSR READ READS BCKGND
08560	08A2	FE	A050		LDX RDATA1
08570	08A5	BC	A064		CPX BEE

08590 08A8 27 03	BEQ	*+5
08590 08AA BD 08D7	JSR	ERROR7
08600 08AD B6 A055 A3	LDA A	ADDR
08610 08B0 81 1F	CMP A	#\$1F
08620 08B2 26 01	BNE	*+3
08630 08B4 39	RTS	
08640 08B5 7C A055	INC	ADDR
08650 08B8 20 B2	BRA	A2
08670 08EA 0D	CERR7	FCB \$D,\$A
08680 08BC 45	FCC	/ERR-TYPE 7 /
08690 08C7 04	FCB	4
08700 08C8 0D	CERR8	FCB \$D,\$A
08710 08CA 20	FCC	/ERR-TYPE 8 /
08720 08D6 04	FCB	4
08730 08D7 CE 08BA	ERROR7	LDX #CERR7
08740 08DA 36	PSH A	
08750 08DB 7E 0620	JMP	ERR1A
08750 08DE CE 08C8	ERROR8	LDX #CERR8
08770 08E1 36	PSH A	
08780 08E2 7E 0647	JMP	ERR2A
08800	* THIS SUBROUTINE SWAPS PLACES WITH THE	
08810	* BACKGROUND PATTERN WORD AND THE TEST	
08820	* WORD LOCATED IN TEE AND BEE. THE CONTENTS	
08830	* OF THE INDEX REGISTER AND XTEMP ARE DESTROYED.	
08840 08E5	SWAP	EQU *
08850 08E5 FE A064	LDX	BEE
08860 08E8 FF A056	STX	XTEMP
08870 08EB FE A062	LDX	TEE
08880 08EE FF A064	STX	BEE
08890 08F1 FE A056	LDX	XTEMP
08900 08F4 FF A062	STX	TEE
08910 08F7 39	RTS	
08930	END	
INBYTE E055		
OUTCH E075		
PDATA1 E07E		
OUT2H E0BF		
OUT4HS E0C8		
OUT2HS E0CA		
INCHAR E1AC		
M1KBUG E0D0		
A1PIAD 8010		
A2PIAD 8014		
B1PIAD 8012		
B2PIAD 8016		
A1PIAC 8011		
A2PIAC 8015		

PAGE 021

PAGE 022

PAGE 023 NCR.2051

B1PIAC 8013	DECS 04CC	E3A 07F1
B2PIAC 8017	CHANGE 04F1	CERR4 07FB
RDATA1 A050	COMM21 0517	CERR5 0809
RDATA2 A051	COMM22 051F	CERR6 0817
VDATA1 A052	COMM23 0531	GALPAT 0825
WDATA2 A053	COMM19 053A	AA 0832
STATUS A054	COMM20 0543	A1 083B
ADDR A055	COMM24 0556	CTEST 0869
XTEMP A056	COMM25 056B	A2 086C
EPWID A066	PTR A06A	A3 03AD
WPWID A068	CURADD A06A	CERR7 08PA
WMODE A06D	COMADD A06B	CERR8 08C8
INIT 0106	CDATA1 A06C	ERROR7 08D7
WINIT 0138	TEE A062	ERROR8 08DE
WRITE 0150	BEE A064	SWAP 08E5
WRITEM1 0157	WAKPAT 0583	
BOKE 0170	LCOP1 0589	TOTAL ERRORS 00000
WONLY 017C	LOOP2 059B	
EONLY 0183	LOOP2A 05B3	
DELAYE 018A	LCOP2B 05BB	
DELAY 0192	LOOP3 05C7	
DEL1 0193	ENDIT 05E2	
RINIT 01A2	BCKGND 05E3	
READ 01BA	BCK1 05EC	
READ1 01C1	BCK2 05FE	
MON2E 01EE	ERROR1 0619	
MON2E2 01F1	ERR1A 0620	
MON2 01FA	ERR1B 0626	
EXMON 0245	ERROR2 0640	
COMM16 0279	ERR2A 0647	
COMM17 0314	ERRCR3 064F	
COMM18 0326	CERR1 0658	
MARCH1 0349	CERR2 0666	
MASES1 0354	CTRR3 0675	
GALFA1 0360	CADD 068B	
WAKPAT1 036C	DISPLAY 0693	
RSTART 0378	DIS1 06AA	
START 0396	DIS2 06AD	
DECIS 03B1	MARCH1 06C6	
H A061	LP1 06D2	
K A050	LP2 06F5	
INHEX 03D7	MASEST 0714	
INHEX1 03DD	PRSET1 0727	
IN1HG 0410	PRSET2 072F	
DELETE 0432	PRSET 0735	
COMM10 043A	VFY 074F	
C1 0445	TEST 0773	
OCRLF 0450	TST2 0787	
COMM11 045B	TST1 07A9	
COMM13 0471	EVEN 07C1	
COMM12 047A	ERROR6 07CA	
COMM14 0480	ERROR4 07CF	
COMM15 048B	ERROR5 07D9	
PCONTL 04A2	E3B 07E1	

Appendix C

Operator's Manual for the NCR 2051 Test System

Appendix C provides the reader with a manual which may be used as a guide for operating the NCR 2051 test system. A more detailed description of the test algorithms and subroutines is contained in Chapter V.

Appendix C

Contents

	Page
Introduction	103
System Power-Up and Initialization Sequence	103
Loading the Test Monitor	105
Entering the Test Monitor	106
Executing MARCH, GALPAT, or WAKPAT Test Programs	107
Executing MASEST Test Program	109
Displaying Contents of an MCR 2051	109
Display/Change Write and Erase Pulse Widths	110
Controlling Output Destination	110
Setting Read-clock Timing	111
Summary	112
Figure C-1. Test Program Directory	113
Table C1. System Initialization Switch Settings	114
Table C2. Pulse Width Table	115
List of References	116

Operating Manual for the NCR-2051 Tester

This manual describes the procedures for using the NCR-2051 MNOS memory tester. This tester is a microcomputer based system which was designed specifically for testing the NCR-2051. In general, the test system will supply all the necessary prompting messages to provide the operator with sufficient information to run the system without referring to the manual. These messages are described herein. The required memory write and erase pulse widths are controlled by hexadecimal (base 16) data inputs from the operator. The times represented by a given input may be obtained from the pulse width table (Table C2).

System Power-up and Initialization Sequence

In order to simplify the power-up sequence, all electrical connections for the different parts of the system are plugged into a common power outlet strip. A single switch located on this power strip may then be used to turn the system on and off. Since there are not enough outlets on the power strip for the videc monitor to be plugged into it, it is necessary to turn the on/off switch (on the monitor) to the on position and check to see that the receiver/monitor switch (located on the right end of the monitor cabinet) is in the monitor position (up).

Verify that the power indicator lights on the microcomputer and the video terminal are on and that the power switch on the cassette interface is in the on position. To clear the video screen of the random characters and initialize the cursor to the lower left corner of the screen, press the "H" key located at the lower right hand corner of the keyboard (not the key for the character 'H'). Next strike the carriage return key (marked "return") and the computer should respond with an asterisk. If this occurs, proceed to the next paragraph. If this is not the case, press the reset button on the microcomputer front panel. If there is still no asterisk, check the local/remote switch on the cassette interface (this switch should be in the remote position). If there is still no asterisk, check all connections between the components of the system to be certain that they are all plugged in securely. Next, compare all switch settings with the list in Table C1 and correct any discrepancies. If the asterisk still does not appear when the reset button is depressed, consult the manuals provided with the microprocessor kit (Ref 1) for additional help in locating the problem.

Loading the Test Monitor

If the test monitor is resident in the microcomputer (i.e. the EPROM board installed) omit this step. After having completed the power-up sequence and obtained the asterisk in response to a carriage return input from the keyboard, it is necessary to set the baud rate switches on the backs of both the terminal and the microcomputer to 300 baud. Set the read select switch (on the cassette interface) to input A or B corresponding to the side to which the tape recorder output is connected. The record and read status switches (also on the cassette interface) should be set in the center (auto) position.

Insert the tape cassette containing the test monitor object code into the cassette recorder and rewind the tape to the beginning (be certain that the tape is inserted with the side (containing the program) facing away from the recorder). It may be necessary to position the manual/auto switch to the manual position in order to rewind the tape (be sure to return this switch to auto when the rewind is completed). Place the recorder in the play mode by depressing the play button (this button must remain down after it is released). The loading process is initiated by typing the "L" character on the keyboard. The cassette tape should begin moving, the read "ready" light should be lighted, and (when the beginning of the recorded program is

reached on the tape) the read "data" light should flicker as the program is loaded into the computer.

When the program is loaded, the computer will stop the tape recorder and turn off the read "ready" light. It will then transmit an asterisk to indicate that the operation is completed. The stop button on the cassette recorder should now be depressed.

Entering the Test Monitor

The 'go to user's program' function of MIKbug (Ref 1) is used to enter the test monitor. Before using this function, it is necessary to load the correct values for the condition code register (located at \$A043) and the program counter (located at \$A048/9). The condition code register should be loaded with \$10 so that interrupts are not allowed. The data which is loaded into the program counter is the starting address of the the test monitor (\$0100 for the version loaded from cassette tape and \$D000 for the EPROM version). The following sequence demonstrates entering the cassette tape version. (Underlined entries are user inputs and 'n' entries are random number computer responses.)

```
*M A043
A043 nn 10
A044 nnReturn
A045 nnReturn
A046 nnReturn
A047 nnReturn
```

A048 nn 01
A049 nn 00
A04A nn
*G

After the user types the G, the test monitor then outputs the directory of all available test programs (See Fig C-1) and waits for a one or two digit input (from the operator) followed by a carriage return. This input represents the number of a program as indicated in the directory. The monitor will then enter the selected program (after printing the name of the program selected). The operation of each program is described in the following paragraphs. At any point in the operation of the test system, when the program is waiting for an input from the operator, the operator may decide to jump back to the test monitor entry point (selected by typing a control C), jump to MIKbug (selected by entering an X), or delete any entry (type control X before the carriage return has been entered).

Executing MARCH, GALPAT, or WAKPAT Test Programs

After entering the test monitor and setting the pulse widths to the desired values, insert a test device into the MUT test socket (pin 1 is located nearest the lever on the zero insertion force socket). Type a "1" followed by a carriage return to enter MARCH (type "3" for GALPAT and "4" for WAKPAT). (See Ref 2 for a description of these

algorithms.) The monitor will respond with "program selected" followed by the name of the program which was selected. The current value of the background test pattern and the test word are then displayed and the option "Go or Re-enter data" offered to the operator. If the patterns are the desired value, then type "G" and the test will commence. If different values are to be used, type "R" and the computer will respond by displaying the instruction "input background test pattern" and wait for the new pattern to be input. The input buffer is zero filled prior to the first input so leading zeroes need not be entered. Only the last four inputs preceding a carriage return are valid, therefore corrections may be made by simply retyping all four digits or using the delete function and re-entering the pattern. The computer then responds with "input testword" and waits for the new testword to be input. The background pattern and test word are again displayed along with the option to "go or re-enter". When the desired values are entered, type "G" to begin program execution. The computer prints "test in progress" as it begins the testing of the device.

All errors (errors meaning that the data which was read does not correspond to that which was written) which are detected will cause an error statement to be printed on the output device (one statement for each error). When the test

is complete, the computer prints "----test complete---" and if no error statements have been printed the device has successfully passed the test. The test monitor will be re-entered and the test directory displayed on the output device.

Executing MASEST Test Program

After entering the test monitor and setting the pulse widths to the desired values, insert a test device (NCR 2051) into the MUT test socket (pin 1 is located nearest the lever on the zero insertion force socket). Type "2" followed by a carriage return to select MASEST (See Ref 2 for a description of the MASEST algorithm). The computer will print "program selected -- MASEST - test in progress" and begin test program execution. All errors will cause an error statement to be printed on the output device (including written data, read data, and the address where the error occurred). When the test is completed, the computer will print "----test complete---" and re-enter the test monitor (printing the test directory).

Displaying Contents of an NCR 2051

After entering the test monitor and inserting a memory device into the MUT test socket, type "D" followed by a carriage return to enter the DISPLAY program. The computer will display the address followed by its contents for each

address of the entire memory (two entries per display line). When all the contents are displayed on the output device, a "?" will be printed and the computer will wait for the operator to type a character (strike any key) before it will return to the test monitor. The test program directory will again be displayed.

Display/Change Write and Erase Pulse Widths

After entering the test monitor, type "FF" followed by a carriage return to enter the pulse widths routine. The computer will print the current contents of the write and erase pulse width registers (four hexadecimal numbers each) and give the operator the option to re-enter the numbers if desired. Type "O" if the current values are correct or type "R" if re-entry is necessary. The instructions to input each pulse width word are printed (See Table C2 for times corresponding to the hexadecimal numbers). When the correct values are displayed, type "O" to re-enter the test monitor and cause the test directory to be displayed.

Controlling Output Destination

The output from the microcomputer system may be routed to one of four output devices which may be connected to the system. A control P is typed to enter the output destination select program. The computer responds with "select desired output device - TER, TTY, PRTR". The

operator then types at least the first two characters of the name of the output device followed by a carriage return (TER = video terminal, TTY = serial port 0, and PRTR = line printer). The name of the output device which was selected will then be printed on the screen of the video terminal. Selecting TTY allows output to either a teletype or the HP-9820A calculator system (this is determined by a switch setting (TTY/HP-9820A) on the back of the microcomputer). The input source remains the video terminal keyboard regardless of the output destination.

Setting Read-Clock Timing

The read-clock pulse width is set using a rotary switch located on the MUT test fixture. There are three positions for this switch that select a different read-clock pulse width. Each position selects a trimpot to be used as one of the timing components for a oneshot multivibrator which generates the read-clock pulse. The three trimpots are calibrated by observing the CLK signal on the MUT fixture with an oscilloscope and adjusting the clock pulse to the desired width (a test program, that reads the MUT, must be executing during the calibration procedure). The times for each switch setting are labeled using adhesive labels and should be replaced each time the pulse width calibrations are changed.

Summary

The NCR 2051 test system is turned on and off with two switches: a switch on the power strip and the power switch on the television monitor. After the power-up and initialization sequence is verified (i.e. the computer responds to a carriage return with an asterisk), the test monitor must be loaded (if the EPROM board is not plugged into the system). The monitor is then entered (using MIKbug) and the desired test routine is selected and executed.

SELECT DESIRED PROGRAM

01	MARCH
02	MASEST
03	GALPAT
04	WAKPAT
0D	DISPLAY
FF	PULSE WIDTHS DISPLAY/CHANGE
?	

Fig. C-1. Test Program Directory

TABLE C1. System Initialization Switch Settings

Switch Name	Position
Microprocessor Switches	
TTY/TERM	TERM
CLOCK (IN/BYPASS)	IN (DOWN)
BAUD RATE	SAME AS TERMINAL
Terminal Switches	
CLOCK (IN/BYPASS)	IN (DOWN)
CURSOR (TOP/BOTTOM)	BOTTOM (DOWN)
LOCAL/COMP	COMP (DOWN)
POWER (ON/OFF)	ON (UP)
SCREEN READ (SCR)	OFF (DOWN)
DATA TRANSFER	POS 4
BAUD RATE	SAME AS MICROPROCESSOR
VIDEO MONITOR	
POWER	ON (CLOCKWISE)
TV/MON	MON (UP)

TABLE C2. Pulse Width Table

Time in milliseconds	Hexadecimal value
----------------------	-------------------

200	4000
100	2000
50	1000
32	0A00
6.4	0200
2	00A0
1	004E

Time in microseconds	Hexadecimal value
----------------------	-------------------

750	0039
500	0025
400	001D
230	0010
200	000D
160	000A
105	0005
92	0004
80	0003
67	0002
55	0001

NOTE:

This table is to be used to look up the hexadecimal value to be entered into the computer to set the erase and write pulse width time delays in the erase/write subroutine. Locate the desired time in the left column, then find the corresponding hexadecimal value from the right column.

List of References

1. M6800 Microprocessor Operating Manual. San Antonio, Texas: Southwest Technical Products Corporation, undated.
2. Robertson, Joel W. Design of a Microprocessor Based System for Testing of the NCR 2051 MNOS Memory. Thesis GE/EE/78-3. Wright Patterson Air Force Base, Ohio: Air Force Institute of Technology, March 1978.

Appendix D

Pulse Width Table

Appendix D provides the reader with a table which is to be used to determine the pulse widths for erase and write operations. The table lists hexadecimal values which are entered using the pulse widths display/change subroutine (See Appendix C) to control the pulse width. The user should first determine the desired pulse width for erase and write and locate the times in the table. The corresponding entry in the table will be a hexadecimal value (on the same line of the table) and should be input to the computer for the pulse width values to be set.

TABLE D1. Pulse Width Table

Time in milliseconds	Hexadecimal value
----------------------	-------------------

200	4000
100	2000
50	1000
32	0A00
6.4	0200
2	00A0
1	004E

Time in microseconds	Hexadecimal value
----------------------	-------------------

750	0039
500	0025
400	001D
230	0010
200	000D
160	000A
105	0005
92	0004
80	0003
67	0002
55	0001

NOTE:

This table is to be used to look up the hexadecimal value to be entered into the computer to set the erase and write pulse width time delays in the erase/write subroutine. Locate the desired time in the left column, then find the corresponding hexadecimal value from the right column.

Appendix E

Flowcharts

Appendix E contains flowcharts for the the test monitor, test algorithms, and test subroutines as follows:

Figure	Subject of Flowchart
E-1	Test Monitor
E-2	MARCH Test Algorithm
E-3	MASEST Test Algorithm
E-4	WAKPAT Test Algorithm
E-5	GALPAT Test Algorithm
E-6	ECKGND Subroutine
E-7	READ Subroutine
E-8	WRITE Subroutine
E-9	Operating System Input Character Routine
E-10	Operating System Output Character Routine

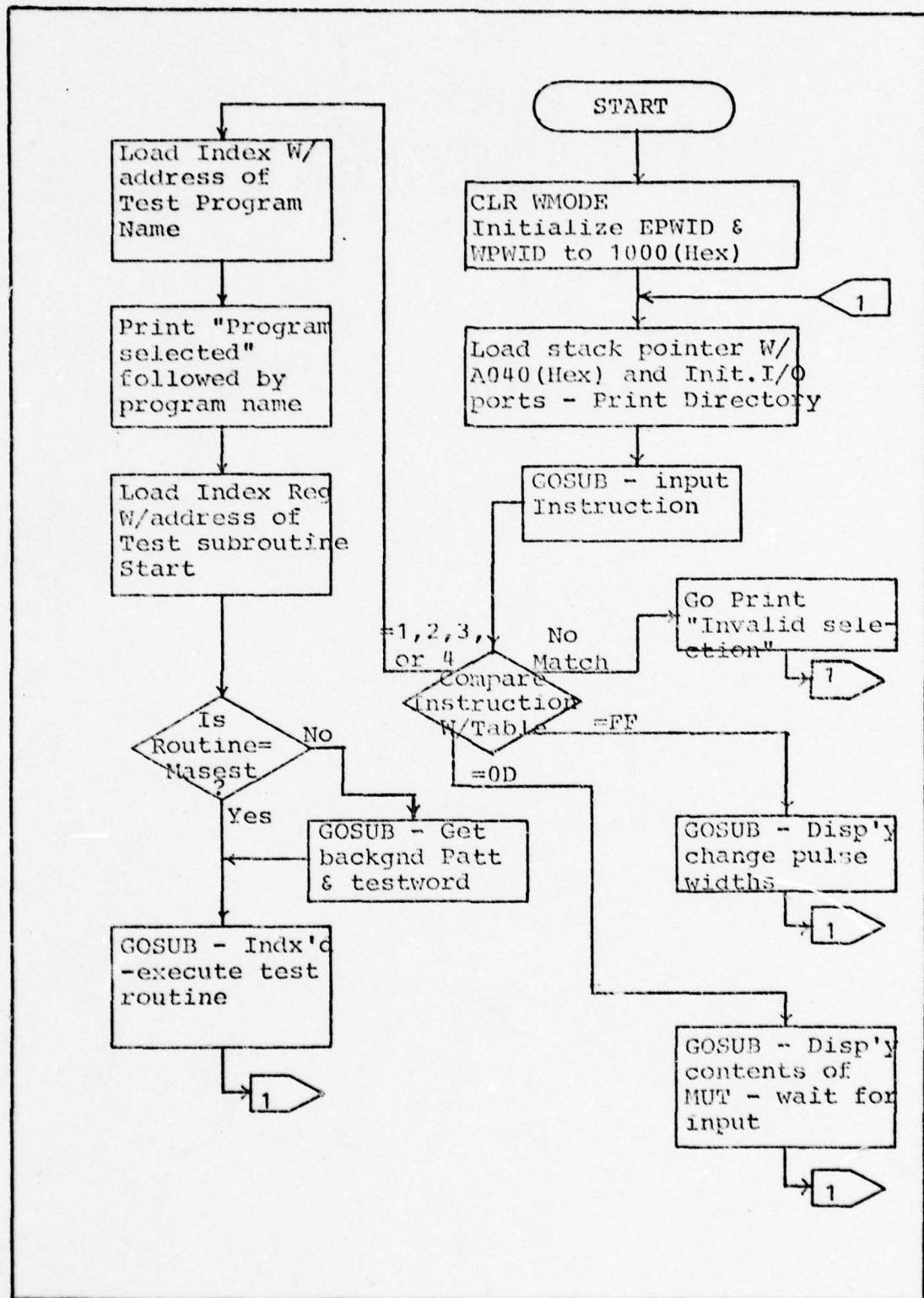


Fig..E-1. Flowchart of Test Monitor

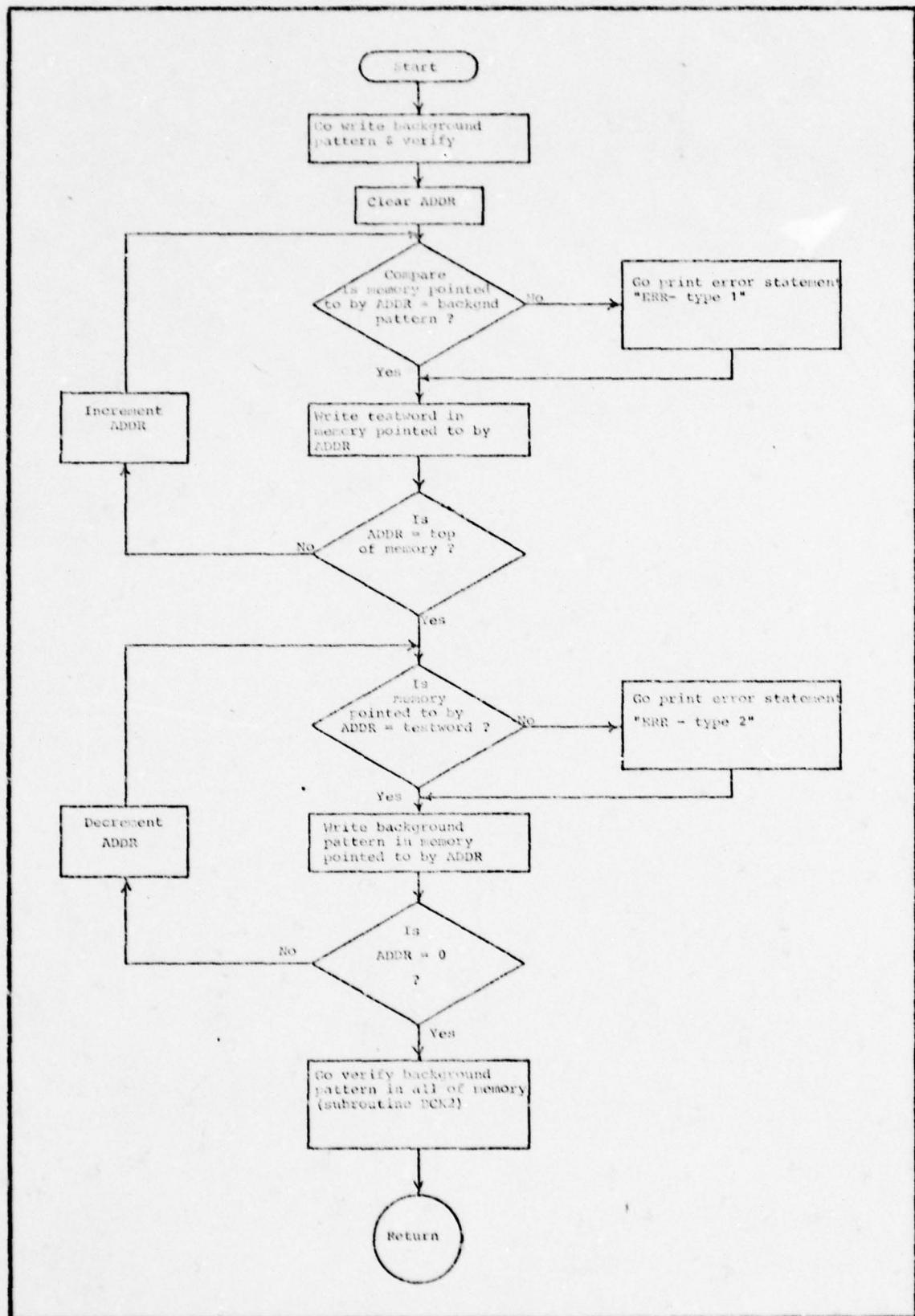


Fig. E-2. Flowchart of MARCH Test Algorithm

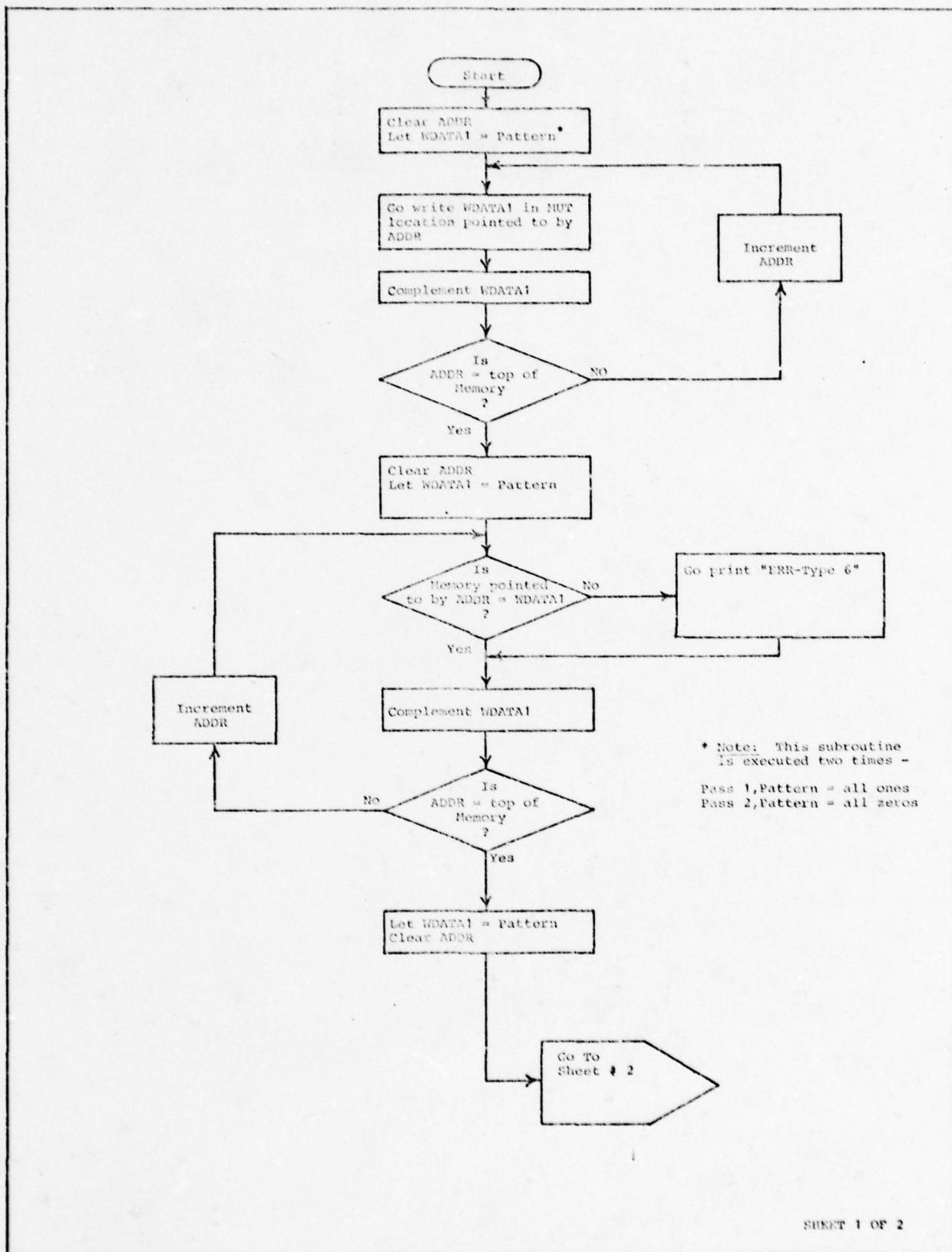


Fig. E-3a. Flowchart of MAEST Test Algorithm

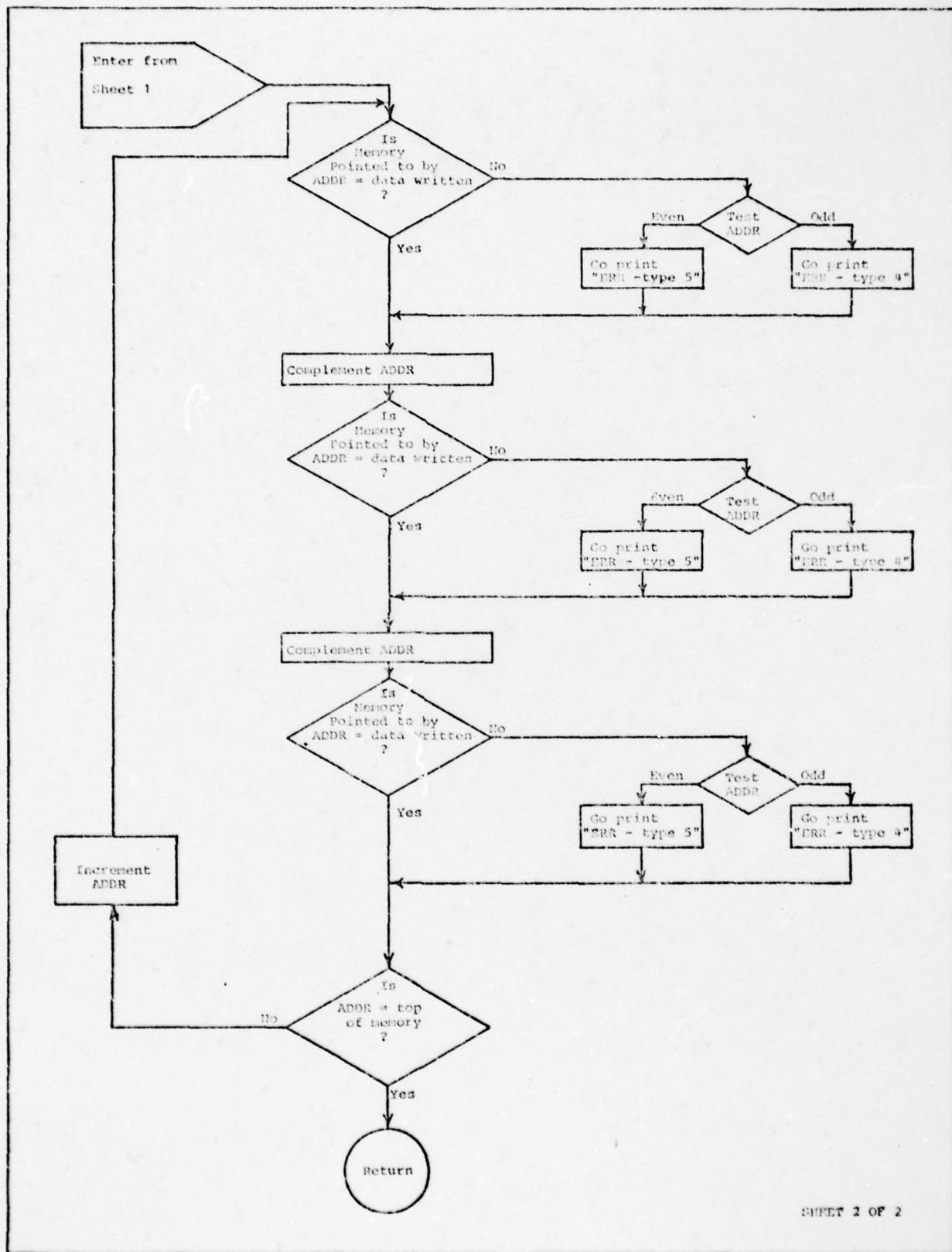


Fig. E-3b. Flowchart of MASEST Test Algorithm

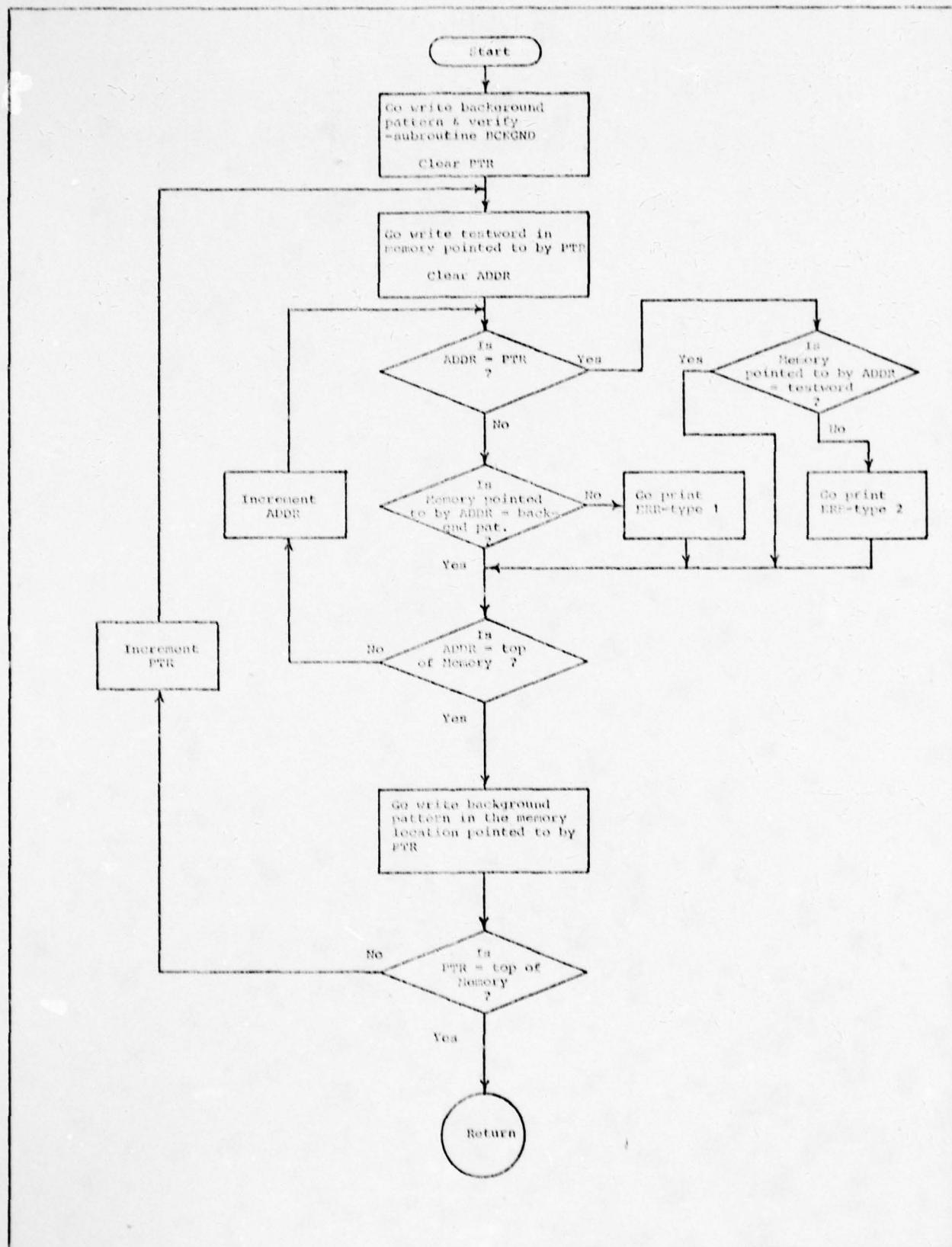


Fig. E-4. Flowchart of WAKPAT Test Algorithm

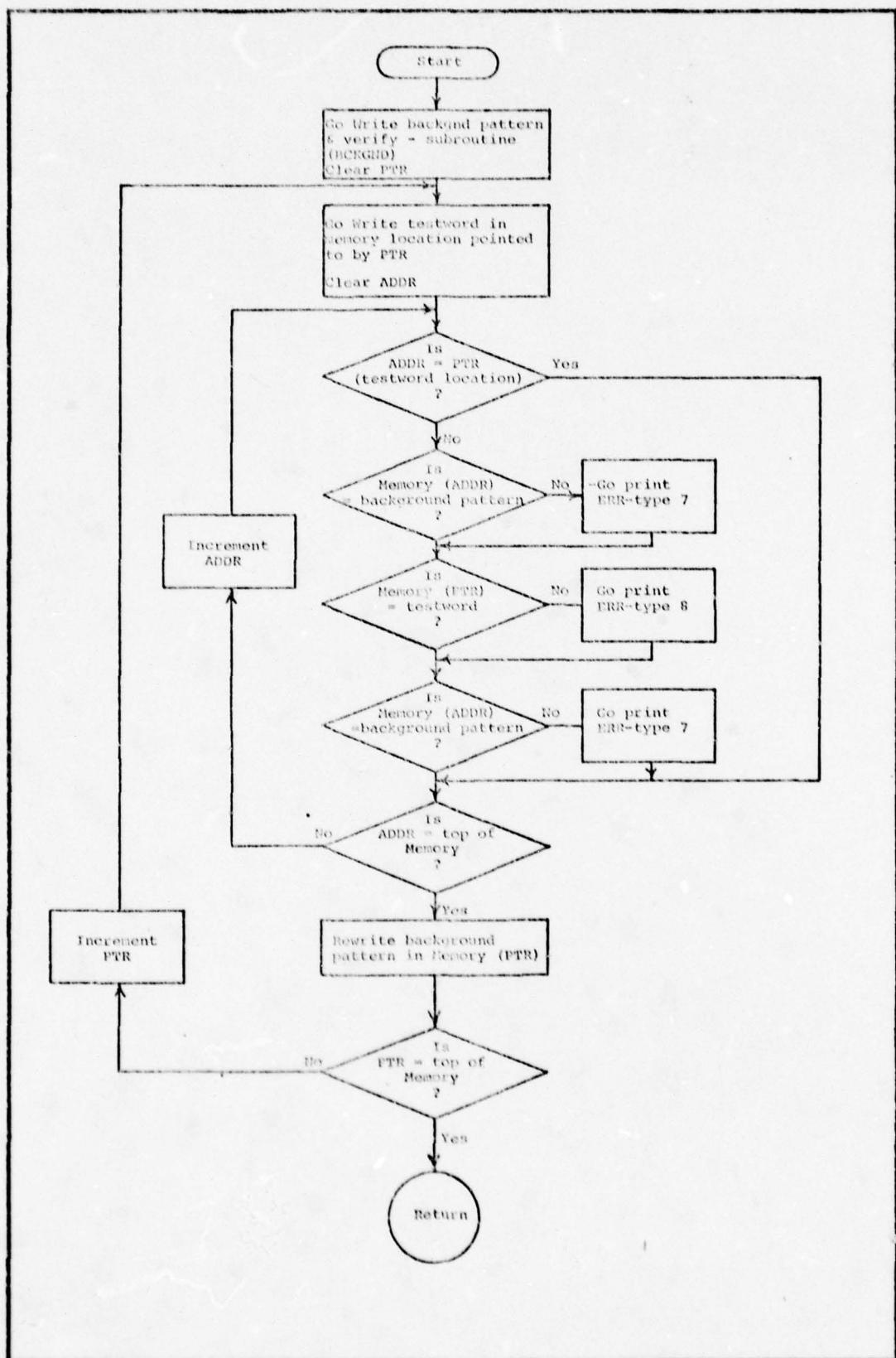


Fig. E-5. Flowchart of GALPAT Test Algorithm

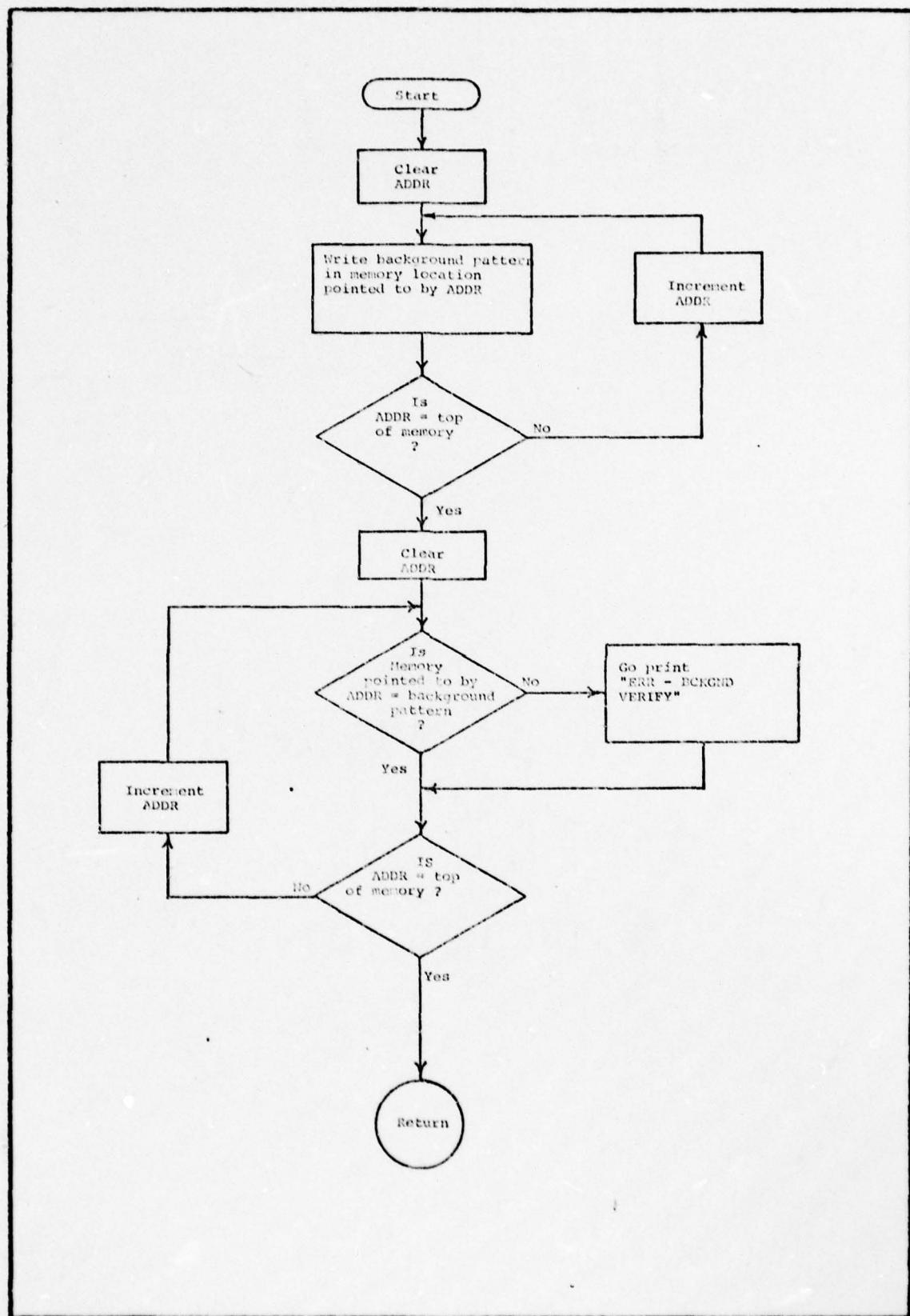


Fig. E-6. Flowchart of BCKGND Subroutine

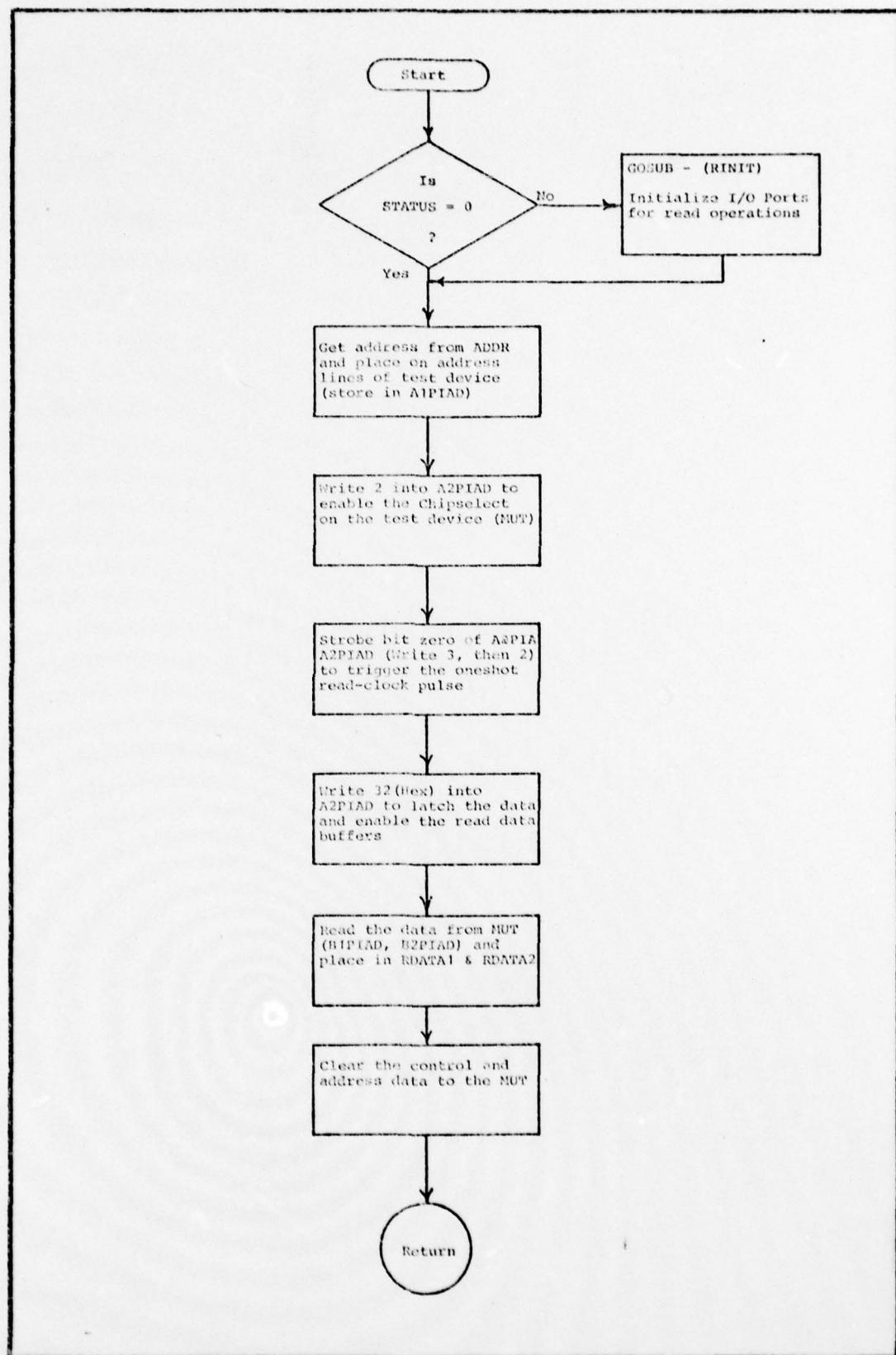


Fig. E-7. Flowchart of READ Subroutine

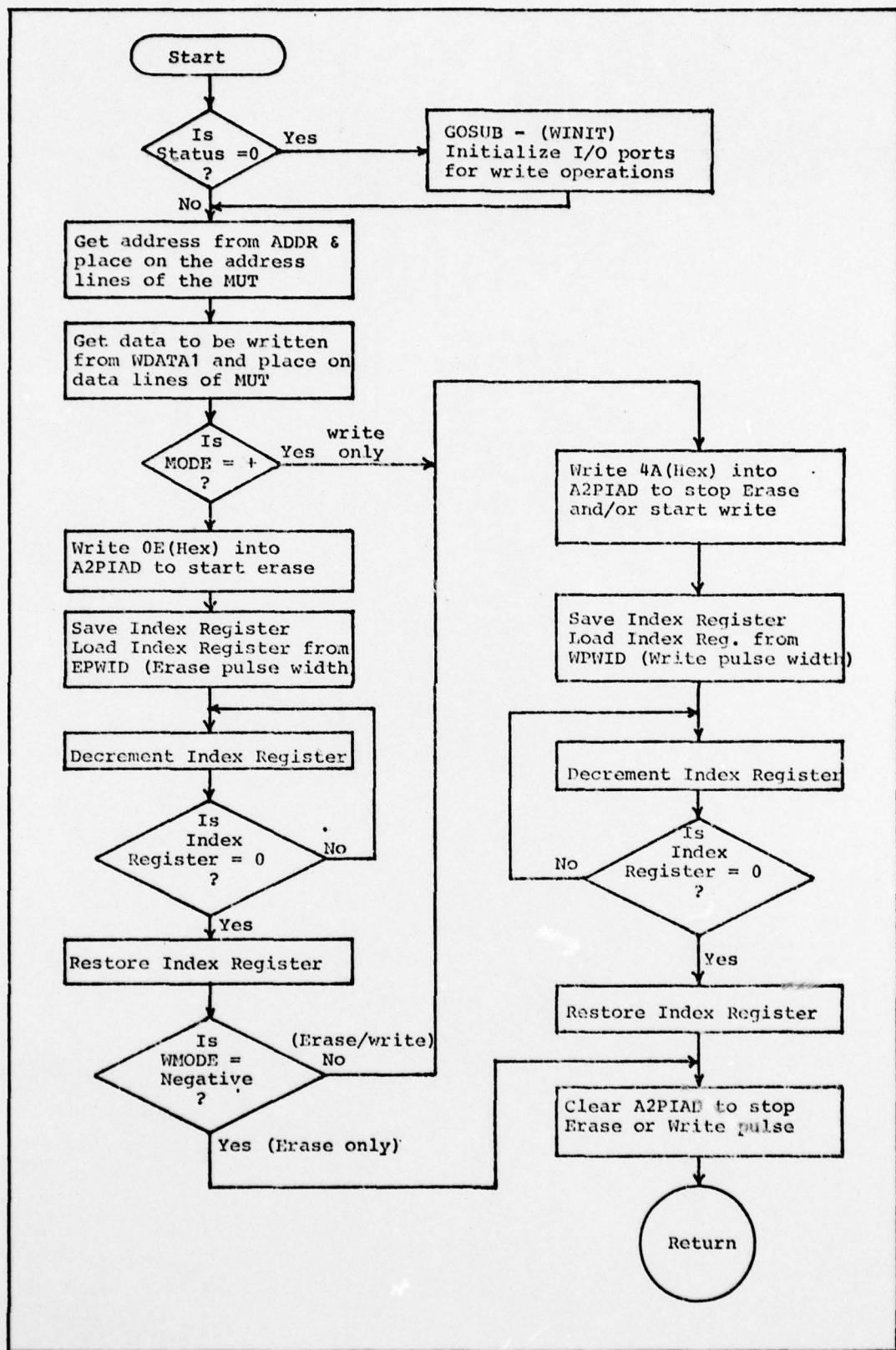


Fig. E-8. Flowchart of WRITE Subroutine

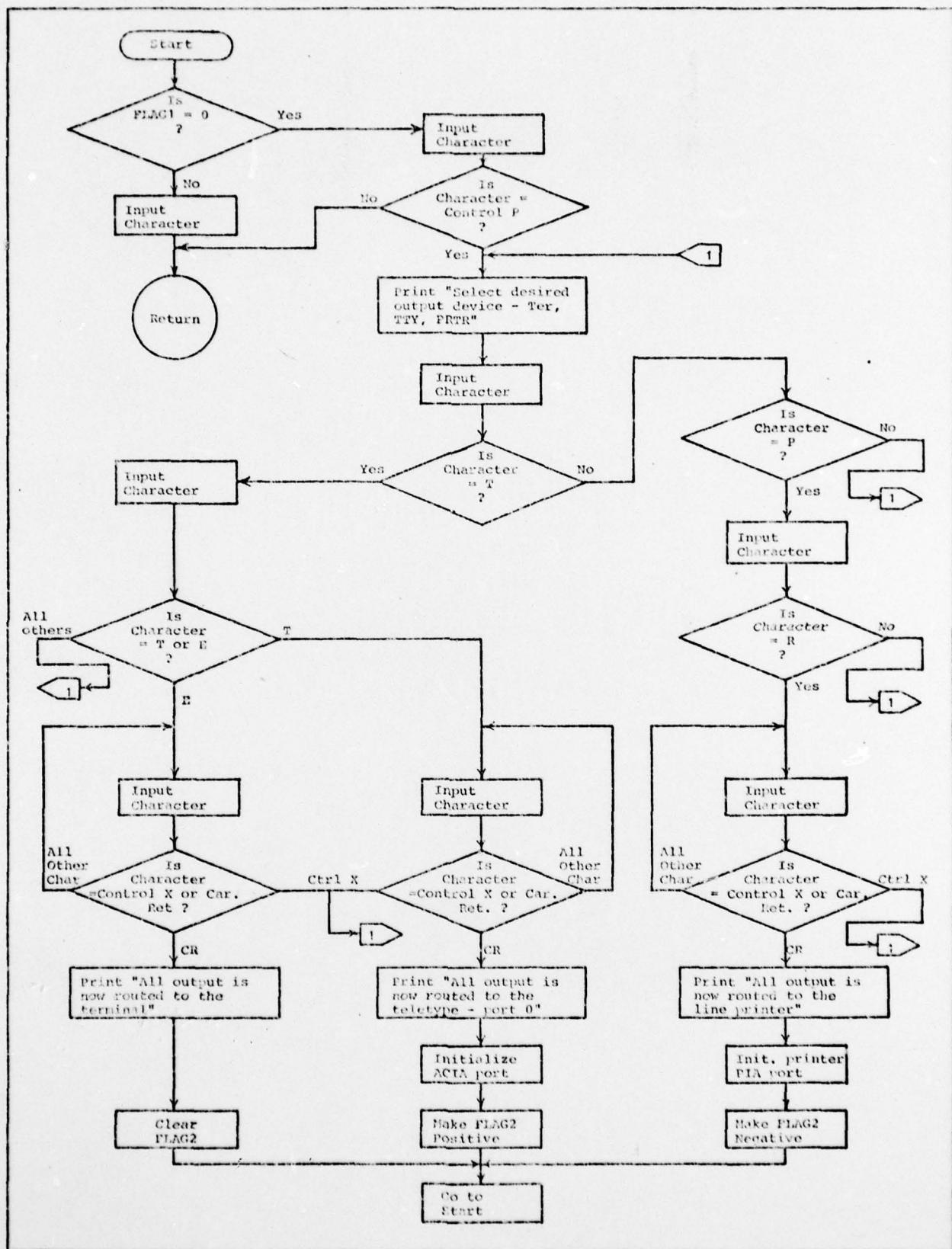


Fig. E-9. Flowchart of Operating System Input Character Routine

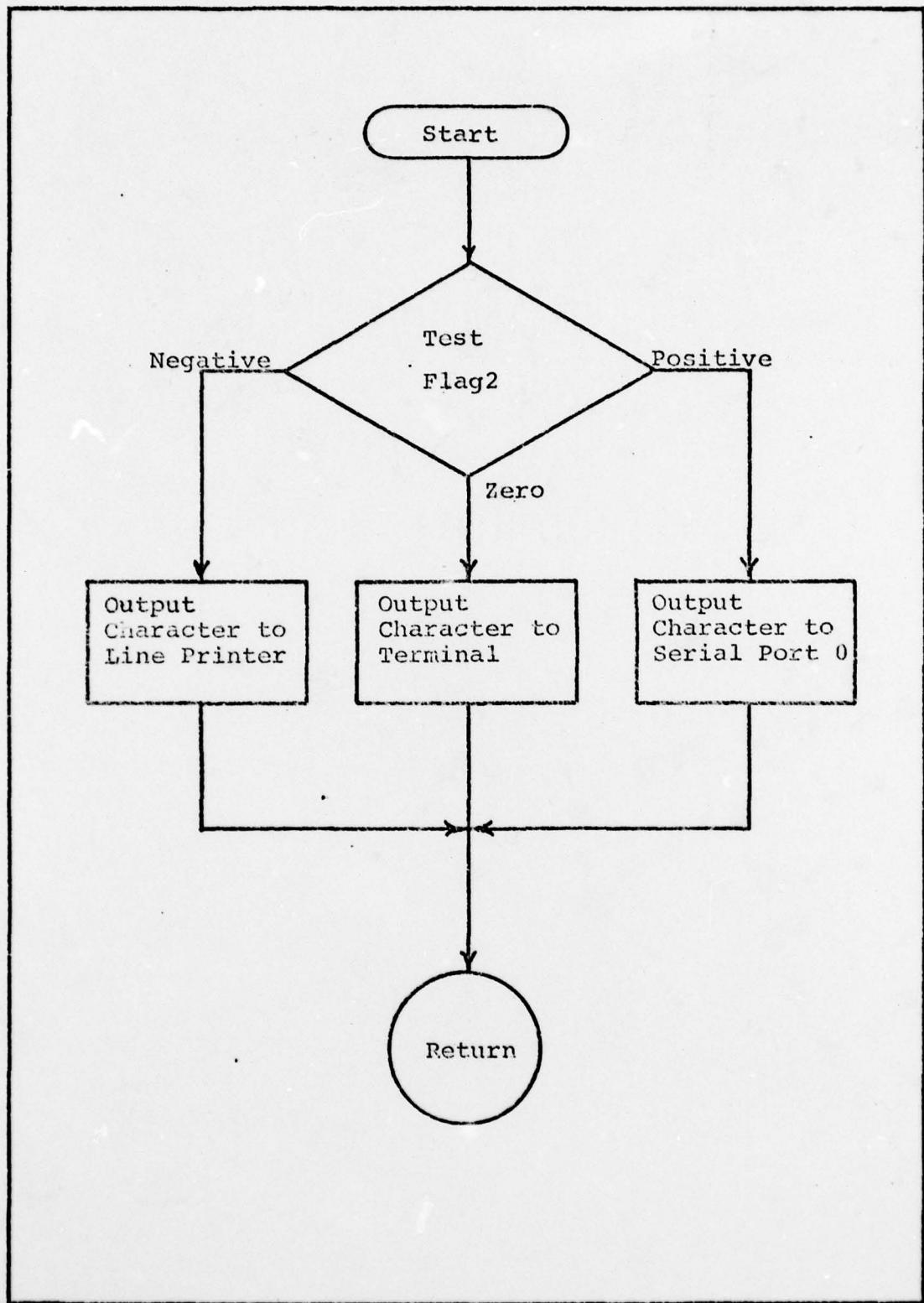


Fig. E-10. Flowchart of Operating System Output Character Routine

Appendix F

NCR 2051 Data Sheets

This appendix contains the data sheets provided by NCR Corporation for the NCR 2051 MNOS memory. These data sheets were reproduced through the courtesy of NCR Corporation.



NCR CORPORATION

PRELIMINARY DATA
512-BIT WAROM™ MEMORY

2051

MICROELECTRONICS DIVISION 8181 BYERS ROAD MIAMISBURG, OHIO 45342

(513) 866-7471
TLX 28-8010 NCRMICRO, MSBG

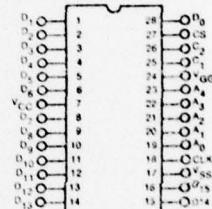
Electrically alterable ROM
MNOS P-channel technology



STANDARD 28 PIN SIDE BRAZE DIP
(1.4 x 0.6 IN.)

- 32 Word x 16 Bits/Word Organization
- 5-Bit Parallel Binary Addressing
- Electrically reprogrammable
- 4.0 μ sec Word Read Access Time
- 40 msec Word Erase Time
- 40 msec Word Write Time
- Minimum Data Retention — 2×10^{11} Read Cycles/Word Before Refresh
- Three State Outputs
- 28 Pin Ceramic Dual In-Line Package
- Unpowered, Nonvolatile Data Storage — 10 Years at 70° C
- Chip-Select, Control, Address, and Data Inputs TTL Compatible

PIN CONFIGURATION



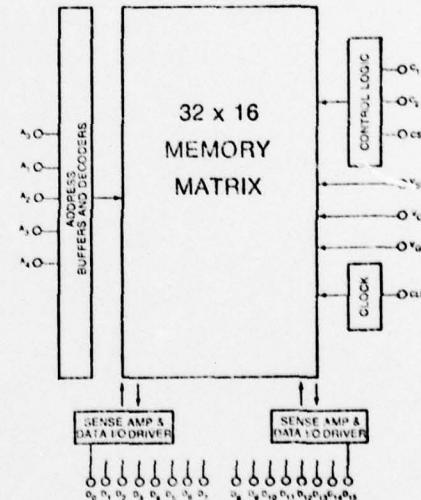
The NCR 2051 is a fully decoded, 32 x 16-bit electrically erasable and reprogrammable ROM utilizing second-generation NCR MNOS epitaxial processing technology.

Data is stored by applying negative writing pulses that selectively tunnel charge into the oxidenitride interface at the gate insulator of the MNOS memory transistors. When the writing voltage is removed, the charge trapped at the interface is manifested as a negative shift in the threshold voltage of the selected memory transistor. Stored data may be accessed a minimum of 2×10^{11} times before refresh is necessary, and is nonvolatile in the unpowered state in excess of 10 years. Although the NCR 2051 is not intended for use as a read/write memory, data can be erased and rewritten up to a maximum of 10^6 times.

All reading, writing, and erasing is accomplished through the use of internal voltage shifting. Hence, high voltage switching and its associated logic are not required for operation.

While the information herein presented has been checked for both accuracy and reliability, NCR assumes no responsibility for either its use or for the infringement of any patents or other rights of third parties, which would result from its use. The publication and dissemination of the enclosed information confers no license, by implication or otherwise, under any patent or patent rights owned by NCR.

FUNCTIONAL BLOCK DIAGRAM



Copyright © 1978 by
NCR Corporation
Dayton, Ohio, U.S.A.
All Rights Reserved
Printed in U.S.A.

PRELIMINARY DATA

2051 512-BIT WAROM™ MEMORY

NCR

ABSOLUTE MAXIMUM RATINGS*

All inputs or outputs relative to V_{SS}	+0.3V to -30V
Operating ambient temperature	0°C to +70°C
Storage temperature	-65°C to +150°C
Soldering temperature of leads (10 seconds)	+300°C

* Stresses above "absolute maximum ratings" may result in damage to the device. Functional operation of devices at the "absolute maximum ratings" or above the recommended operational limits stipulated elsewhere in this specification is not implied.

RECOMMENDED OPERATING CONDITIONS, $T_A = 0^\circ C$ TO $70^\circ C$

Symbol	Parameter	Power Supply Requirements					
		TTL			CMOS		
V_{GG}		-20 ± 1.5V			24 ± 1.5V		
V_{SS}		15.0 ± 0.5V			+10 ± 1.0V		
V_{CC}		0			0		
Symbol	Parameter	Erase	Write		Read		
		Min	Typ	Max	Min	Typ	Max
V_{CKH}	Clock input high	$V_{SS} - 1.5$	V_{SS}	$V_{SS} + 3$	$V_{SS} - 1.5$	V_{SS}	$V_{SS} + 3$
V_{CKL}	Clock input low	$V_{SS} - 15$	V_{CC}	$V_{SS} - 4.0$	$V_{SS} - 15$	V_{CC}	$V_{SS} - 4.0$
V_{C1H}, V_{C2H}	Mode control high	$V_{SS} - 1.5$	V_{SS}	$V_{SS} + 3$	$V_{SS} - 1.5$	V_{SS}	$V_{SS} + 3$
V_{C1L}, V_{C2L}	Mode control low	$V_{SS} - 15$	V_{CC}	$V_{SS} - 4.0$	$V_{SS} - 15$	V_{CC}	$V_{SS} - 4.0$
V_{CSH}	Chip select high	$V_{SS} - 1.5$	V_{SS}	$V_{SS} + 3$	$V_{SS} - 1.5$	V_{CC}	$V_{SS} + 3$
V_{CSL}	Chip select low	$V_{SS} - 15$	V_{CC}	$V_{SS} - 4.0$	$V_{SS} - 15$	V_{CC}	$V_{SS} - 4.0$
V_{AH}	Word address high	$V_{SS} - 1.5$	V_{SS}	$V_{SS} + 3$	$V_{SS} - 1.5$	V_{SS}	$V_{SS} + 3$
V_{AL}	Word address low	$V_{SS} - 15$	V_{CC}	$V_{SS} - 4.0$	$V_{SS} - 15$	V_{CC}	$V_{SS} - 4.0$
V_{DH}	Data I/O high	$V_{SS} - 1.5$	V_{SS}	$V_{SS} + 3$	$V_{SS} - 1.5$	V_{SS}	$V_{SS} + 3$
V_{DL}	Data I/O low	$V_{SS} - 15$	V_{CC}	$V_{SS} - 4.0$	$V_{SS} - 15$	V_{CC}	$V_{SS} - 4.0$

STATIC ELECTRICAL CHARACTERISTICS, $T_A = 0^\circ C$ TO $70^\circ C$ NO EXTERNAL LOADS EXCEPT AS NOTED

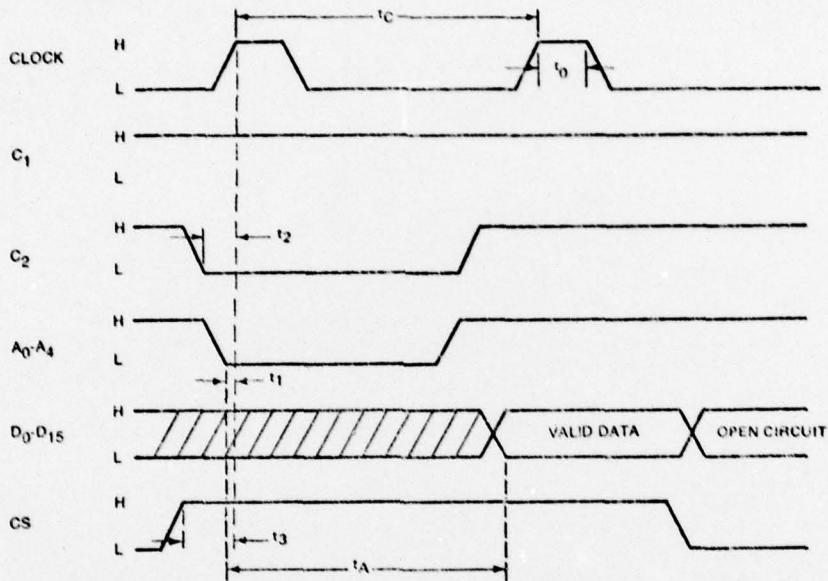
Symbol	Parameter	Conditions		Min	Typ	Max	Unit
		All Pins at V_{SS} Unless Noted					
I_{IN}	Input leakage current (Pins 18-23, 25-27)	$V_{IN} = V_{SS} - 15V$				2.0	μA
I_{OUT}	Output leakage current	$V_{OUT} = V_{SS} - 15V, V_{CS} = LOW, V_{GG} = V_{SS} - 34V$				-1.0	μA
I_R	V_{GG} supply current, Reed mode	$V_{GG} = V_{SS} - 34V$				-12	mA
I_W	V_{GG} supply current, Write mode	$V_{GG} = V_{SS} - 34V$	Outputs open		-7	8	mA
I_{OH}	Data output high current	$V_{OH} = 3.5V$					mA
I_{OL}	Data output low current	$V_{OL} = 0V$		-1.0			mA
V_{OH}	Data output high voltage	$C_L = 100 \text{ pf}$		+1.6			V
V_{OL}	Data output low voltage	$C_L = 100 \text{ pf}$		$V_{SS} - 1.5$			V
t_s	Unpower nonvolatile data storage	Following minimum write conditions		10			Years

CAPACITANCE WITH ALL PINS GROUNDED, $f = 1$ MHz

Symbol	Parameter	Min	Typ	Max	Unit
C_A	Address and clock input capacitance		5	7	pf
C_D	Data input/output capacitance		6	10	pf
C_C	Mode control and chip select capacitance		5	7	pf

NCR**PRELIMINARY DATA****512-BIT WAROM™ MEMORY****2051**READ CYCLE CHARACTERISTICS, $T_A = 0^\circ\text{C}$ to 70°C

Symbol	Parameter	Min	Typ	Max	Unit
t_0	Clock pulse width	2.0		20	μsec
t_c	Cycle time ($t_0 = 2.0 \mu\text{sec}$)	4.0			μsec
t_A	Access time ($t_0 = 2.0 \mu\text{sec}$)			4.0	μsec
t_1	Address - clock separation	0			μsec
t_2	Select - clock separation	0			μsec
t_3	Mode - clock separation	0			μsec
t_R	Input Rise Time	0		100	μsec
t_F	Input Fall Time	0		100	μsec



MODE CONTROL LOGIC

C ₁	
OV	+5V
C ₂	OV Write Read +5V Erase Read

Mode	Function
Read	Addressed data read after CLK pulse.
Write	Input data written, CLK pulse arbitrary.
Erase	Stored data is erased at addressed location.

NCR**PRELIMINARY DATA****512-BIT WAROM™ MEMORY****2051****ERASE CYCLE CHARACTERISTICS, $T_A = 0^\circ\text{C}$ TO 70°C**

Symbol	Parameter	Min	Typ	Max	Unit
t_E	V_E Erase pulse width	40		200	msec
t_0	Address Select - Erase Delay	0			nsec
t_1	Mode Select - Erase Delay	0			nsec

WRITE CYCLE CHARACTERISTICS, $T_A = 0^\circ\text{C}$ TO 70°C

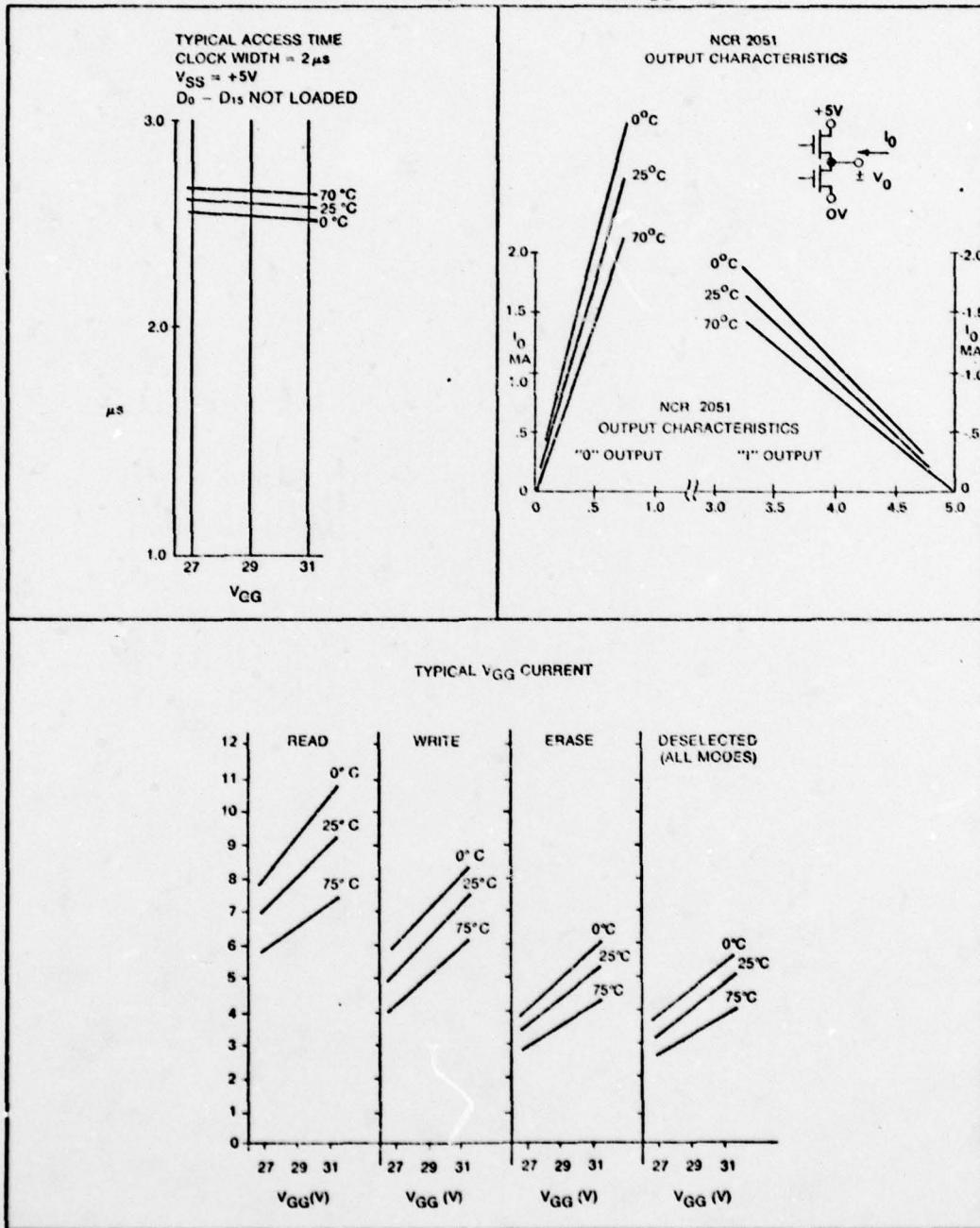
Symbol	Parameter	Min	Typ	Max	Unit
t_W	Write pulse width	40		200	msec
t_0	Address Select - Write Delay	0			nsec
t_1	Mode Select - Write Delay	0			nsec
t_2	Data Set - Write Delay	0			nsec

NCR

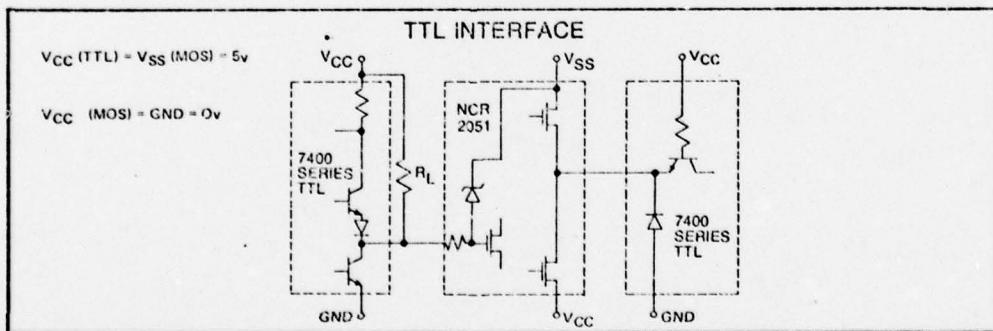
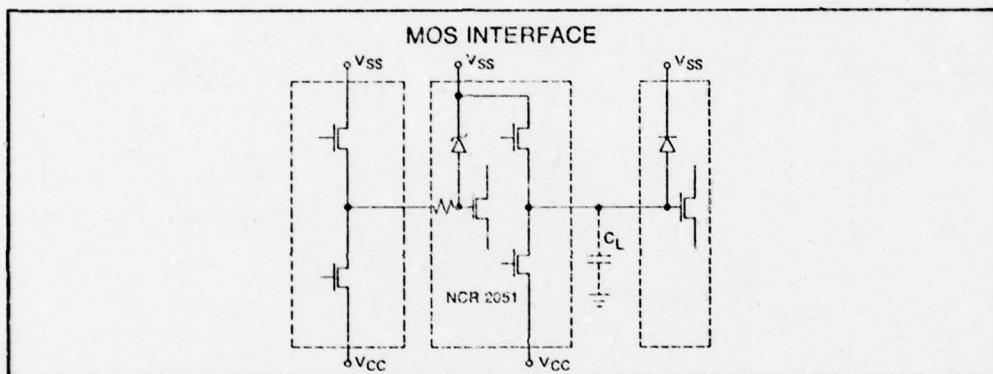
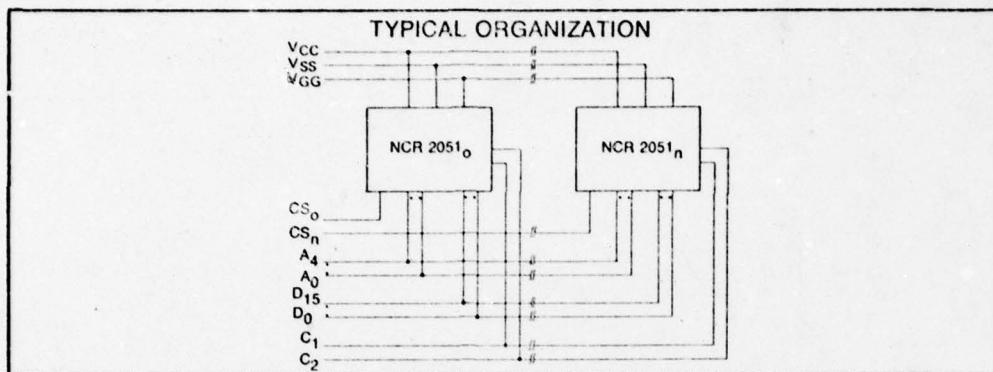
**PRELIMINARY DATA
512-BIT WAROM™ MEMORY**

2051

TYPICAL OPERATING CHARACTERISTICS, $T_A = 0^\circ\text{C}$ TO 70°C , $V_{SS} = 5\text{V}$



PRELIMINARY DATA



NCR
NCR CORPORATION

MICROELECTRONICS DIVISION

1176 06 2/78

512-BIT WAROM™ MEMORY

2051

8181 Byers Road
Miamisburg, Ohio 45342
(513) 866-7471
TLX 28-3010 NCRMICRO, MSBG

Vita

Joel W. Robertson was born on 2 December, 1944 in Wellington, Texas. He graduated from Mirabeau B. Lamar Senior High School in Houston, Texas in June, 1963. After completing two years of college at Howard County Junior College in Big Spring, Texas he enlisted in the U.S. Air Force on 22 January, 1968. After spending three years as an aircraft radio repairman (including one year in South Vietnam) he was accepted into the Airman Education and Commissioning Program. He was assigned to Texas Technological University and received a Bachelor of Science in Electrical Engineering in May, 1973. On 4 September, 1973, he completed the USAF School of Military Science, Officer and was commissioned a second lieutenant in the USAF. The next three years were spent at the Air Force Weapons Laboratory at Kirtland AFB, New Mexico. He participated in power supply development projects for high energy lasers including the first Airborne Laser Laboratory. In August, 1976 he was assigned to the Air Force Institute of Technology.

Permanent Address: 2816 W. Shandon

Midland, Texas 79701

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER GE/EE/78-7	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) Design of a Microprocessor-Based System for testing of the NCR 2051 MNOS Memory		5. TYPE OF REPORT & PERIOD COVERED MS Thesis
7. AUTHOR(s) Joel W. Robertson Captain USAF		6. PERFORMING ORG. REPORT NUMBER
9. PERFORMING ORGANIZATION NAME AND ADDRESS Air Force Institute of Technology (AFIT/EN) Wright-Patterson AFB, Ohio 45433		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS Air Force Avionics Laboratory (AFAL/DHE) Wright-Patterson AFB, Ohio 45433		12. REPORT DATE March 1978
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)		13. NUMBER OF PAGES 138
		15. SECURITY CLASS. (of this report) UNCLAS
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE
16. DISTRIBUTION STATEMENT (of this Report) Approved for public release; distribution unlimited.		
17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)		
18. SUPPLEMENTARY NOTES Approved for public release; IAW AFR 190-17 JERRAL F. GUESS, Captain, USAF Director of Information		
19. KEY WORDS (Continue on reverse side if necessary and identify by block number)		
20. ABSTRACT (Continue on reverse side if necessary and identify by block number) The U.S. Air Force has begun using MNOS memories in ultrahigh frequency radios to provide nonvolatile storage of preset frequencies. The NCR 2051 is one such memory and it is necessary that the USAF have the capability to perform acceptance testing of these devices as well as to economically perform laboratory tests which may be very time consuming. This report develops a microprocessor-based computer system which will provide the necessary capabilities economically. (Continued on reverse)		

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE(When Data Entered)

(Continued from block 20)

The design of the Motorola M6800-based system is presented with both hardware and software considerations. The development decisions are discussed and a user's manual is provided. Complete assembly language software listings, which realize the acceptance testing requirements, are included. Flowcharts for all test algorithms and schematic diagrams for all interface circuits are also provided.

A

9

3

A

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE(When Data Entered)